UNCLASSIFIED

Document no. TRAC-M-TR-13-004
28 September 2012

Social Impact Model (SIM) Transition

Technical Report

TRADOC Analysis Center - Monterey
Naval Postgraduate School
700 Dyer Rd., Rm WA-180
Monterey, California 93943

This study cost the
Department of Defense approximately
$411,000 expended by TRAC in
Fiscal Year 2012.
Prepared on 20121001
TRAC Project Code # 640

UNCLASSIFIED

This page intentionally left blank.

Document no. TRAC-M-TR-13-004
28 September 2012

Social Impact Model (SIM) Transition

Technical Report

LTC Jason Caldwell
Mr. Leroy "Jack" Jackson
Mr. Harold Yamauchi
Mr. John Ruck
MAJ Thomas Deveans
Ms. Kristen Clark

TRADOC Analysis Center - Monterey
Naval Postgraduate School
700 Dyer Rd., Rm WA-180
Monterey, California 93943

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
01-10-2012	Technical Report	October 2011 - September 2012

4. TITLE AND SUBTITLE
Social Impacts Module (SIM) Transition

5a. CONTRACT NUMBER

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
N/A

6. AUTHOR(S)
LTC Jason C. Caldwell, Mr. Leroy "Jack" Jackson, Mr. Harold Yamauchi, Mr. John Ruck, MAJ Thomas Deveans, Ms. Kristen Clark

5d. PROJECT NUMBER
TRAC Project Code 640

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army TRADOC Analysis Center - Monterey
Naval Postgraduate School
700 Dyer Road
Monterey, CA 93943-0692

8. PERFORMING ORGANIZATION REPORT NUMBER
TRAC-M-TR-13-004

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Methods and Research Office
TRADOC Analysis Center (TRAC)
Fort Leavenworth, KS

10. SPONSOR/MONITOR'S ACRONYM(S)
MRO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
TRAC-M-TR-13-004

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

13. SUPPLEMENTARY NOTES
Findings of this report are not to be construed as an official Department of the Army (DA) position.

14. ABSTRACT
The Social Impact Module (SIM) is a discrete event simulation developed and implemented using Java SimKit Libraries. Fiscal year 2012 guidance from the Director of the Training and Doctrine Command (TRADOC) Analysis Center (TRAC) stated that SIM would be transitioned from TRAC-Monterey (MTRY) to TRAC-White Sands Missile Range (WSMR). The transition occurred over a series of milestones than began with defining requirements and ended with the delivery of three software versions. TRAC-WSMR, the primary stakeholder, and TRAC-MTRY, the model provider, worked in concert to ensure development adhered to written specifications. At the conclusion of SIM Transition, TRAC-WSMR owned SIM, a fully functional model that simulates population responses, key leader engagements, and social network activity. Throughout FY12, developers improved model functionality, stabilizing the models for use in TRAC's Irregular Warfare Tactical Wargame (IW TWG). Through rigorous testing, the team verified that model outputs are traceable back to initialization data in the scenario file. Additionally, model documentation ensured that SIM is explainable by the analyst employing the model for the TWG use case.

15. SUBJECT TERMS
Social Impacts Module (SIM); Discrete Event Simulation (DES); TRAC Irregular Warfare Tactical Wargame (IW TWG)

16. SECURITY CLASSIFICATION OF: Unclassified			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Jason C. Caldwell
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(include area code)*
Unclassified	Unclassified	Unclassified	UU	378	831-656-3086

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

This document provides an overview for the transition of the Social Impact Module (SIM). The transition occurred over a series of milestones than began with defining requirements and ended with the delivery of three software versions. TRADOC Analysis Center (TRAC) White Sands Missile Range (WSMR), the primary stakeholder, and TRAC-Monterey (MTRY), the model provider, worked in concert to ensure development adhered to written specifications. At the conclusion of SIM Transition, TRAC-WSMR owns SIM, a fully functional model that simulates population responses, key leader engagements, and social network activity. In addition, WSMR can employ the model for the Irregular Warfare (IW) Tactical Wargame (TWG) use case. After clearly defining a study question and conducting a measurement space workshop, WSMR will be able to design a scenario file for SIM that will provide outputs necessary to stimulate the players in future IW TWG.

The Social Impact Module version 1.0 is comprised of the Cultural Geography (CG) model and Nexus Network Learner (NNL) models. CG is an agent-based model of the operational environment based on doctrine and social theory. CG consists of agents (simulated people) interacting with each other in the conflict environment and responding to wargame player actions within the environment. Each agent is defined by a set of demographic dimensions that collectively shape the agent's beliefs, values, interests, stances on issues, and behaviors. The narrative paradigm is the underlying social theory upon which narrative identities are developed from data to form agent beliefs, values, and interests. The narrative paradigm primarily manifests itself in the data (scenario) development process; however, it also forms the foundation for how an agent perceives events in the simulation.

The key design change in SIM 2.0 is the integration of key leader and social networks into the population model. Previously SIM utilized Nexus Network Learner for key leaders and social networks. Augustine Consulting Incorporated (ACI) contractors implemented Nexus using Java Repast libraries. The reliance on two separate models in SIM was not only inefficient; it required additional coordination, configuration management, and contract dollars to maintain. The transition team's intent behind SIM 2.0 was to consolidate the capabilities of both models into a single, Java SimKit-based discrete event simulation. The resulting model reduced complexity in scenario design, decreased SIM execution time, eliminated the need for communication between two separate models, and reduced reliance on contractor support.

During SIM 1.0 stabilization efforts, the transition team began exploring alternatives to Bayesian Belief Network modeling techniques. Immediately, the team identified Markov Chains as another method of modeling discrete state probabilities. Although a Markovian approach seemed appropriate, the team realized that it did not simplify data requirements or minimize SME elicitation. As a result, the team continued investigating and evaluating other potential methods. Fortuitously, TRAC-MTRY worked on a project, occurring concurrent to SIM Transition, sponsored by the Center for Army Analysis (CAA). This project's name was Africa Knowledge, Data Source, and Analytic Effort (KDAE) Exploration. Part of the Africa KDAE research developed a methodology and built a proof-of-principle scenario in a specific region or country in the United States Army (USA) Africa Command (AFRICOM) area of responsibility (AOR) for use in future IW TWGs using Factor Analysis and Generalized Linear Models (GLM).

The Social Impact Module version 3.0's data development methodology for population modeling changes significantly in this version when compared to version 1.0 or 2.0. The scenario file contains eight (8) fewer worksheets, removing many of the belief and issue related input because Bayesian Belief Networks are no longer utilized for agent issue stances and OAB. SIM 3.0 still contains no loss in functionality when representing key leaders and social networks.

Extensive testing produced a list of recommended practices for model settings. The four primary recommendations for the use of SIM discovered during SIM Transition are:

- Use a discount factor (λ) of 0.01. The discount factor has a significant effect on how long agents remember good and bad events. A lower discount factor ensures that they have a longer "memory", and will result in better and more rational agent behavior over time.
- Population stereotypes should have around 100 survey respondents per agent stereotype prototype. This results in better underlying data for the Bayesian Belief Networks, and is likely to provide more evidence for all combinations in the conditional probability table.
- Avoid using effects data that centers around 50% for any event. This relegates the effects of scenario events to a coin flip, resulting in poor output data from the model.
- Use fewer events or bin similar events to minimize effects in the model. The use of hundreds of events that each carry an effect dilutes the impact of each event and adds unnecessary complexity to the model.

In addition to these four recommendations, the following list outlines other recommendations and best practices for the use of SIM:

- SIM is designed to run less than 2-years of simulated time. It is very capable of running 100-years; however, the development team strongly discourages this type of use. The demographic dimensions modeled in SIM are static, so agents do not get older, change their political views or otherwise "grow" out of a given stereotype.
- SIM works best at the tactical level (Brigade and below). The .
- Scenario designers should pay close attention to the effects on population stereotypes received from SME. One of the primary recommendations was avoiding 50-50% data, but there is another significant issue to guard against. If the effect is less than the initial value (%) issue stance, the issue stance can only decrease, even if it is viewed

positively. This is rare, but happened during testing when the team used extreme issue stances of 99% and 100% adequate. Agents will always be "disappointed" because the effect of a positive action is not as great as their instantiation in the model. The opposite is true about negative agents and negative results. If the effect is greater than the initial issue stance, the issue stance will only increase, even if it has a very low effect.

- SIM is good at modeling issue stances. It can model OAB, but often survey questions do not ask about positive passive, negative active, etc. The questions ask about a person's opinion on the issues. If OABs continue to be part of the IW TWG, consider finding data sources that ask questions specific to the way OABs will be modeled or create the survey within TRAC for modeling the OAB. Other alternatives include using the counter system described in SIM 2.0 and having SME determine if an event will have a positive, negative or neutral effect.

- Use Factor Analysis techniques explored as part of SIM 3.0 development to determine the issues that matter to a modeled population is highly encouraged. Instead of determining a priori what the issues are and forcing population opinion into those bins, use Factor Analysis to allow the data to tell you what is important to the people of a region. These techniques can provide the data needed for SIM 2.0. The design team proved this process works when testing the KDAE data in SIM 2.0.

- The best use of SIM might be to combine the best of different versions. The development team did not have the time or resources to build and test a hybrid configuration; however, SimKit modules are interchangeable. Minor modifications to the SIM code will allow the WSMR contract team could experiment with these possibilities.

- Conducting a calibration exercise before the the next IW TWG is absolutely essential to getting the model results desired by the TWG team. SIM is extremely flexible, and by doing slight modifications to the scenario file, most results can be achieved.

LIST OF ACRONYMS AND ABBREVIATIONS

AAR	After Action Review
ACI	Augustine Consulting Incorporated
AFRICOM	US Army Africa Command
ANOVA	Analysis of Variance
AOR	Area of Responsibility
BBN	Bayesian Belief Network
BVI	Beliefs, Values, and Interests
CAA	Center for Army Analysis
CG	Cultural Geography
CoIST	Company Intelligence Support Team
CK	Critical Knowledge
CM	Configuration Management
CPT	Conditional Probability Table
CSV	Comma Separated Value
DaViTo	Data Visualization Tool
DCMP	Data Collection Management Plan
DES	Discrete Event Simulation
DISA	Defense Information Systems Agency
DOE	Design of Experiment
DP	Design Point
DVD	Digital Versatile Disc
EDA	Exploratory Data Analysis
EL	Exploratory
FMC	Full Mission Capable
GLM	Generalized Linear Models
GUI	Graphical User Interface
HITL	Human in the Loop
HUMINT	Human Intelligence
IMINT	Imagery Intelligence
IPR	In-progress Review
IPT	Integrated Product Team
IW	Irregular Warfare
KDAE	Knowledge, Data Source, and Analytic Exploration
KLE	Key Leader Engagement
MASINT	Measurement and Signature Intelligence
MmAWG	Methods, models, and Analysis Working Group
MOVES	Modeling, Virtual Environments, and Simulations
MS	Masters of Science
N	Neutral (shorthand OAB abbr.)
NA	Negative Active (shorthand OAB abbr.)
NIPR	Non-Classified Internet Protocol Router
NP	Negative Passive (shorthand OAB abbr.)

OAB	Observed Attitudes & Behaviors
OE	Operational Environment
OR	Operations Research
PA	Positive Active (shorthand OAB abbr.)
PAVE	Planning, Adjudication, Visualization Environment
POC	Point of Contact
PoP	Proof of Principle
PP	Positive Passive (shorthand OAB abbr.)
RACE	Rapid Access Computing Environment
RL	Reinforcement Learning
SA	Situational Awareness
SC	Socio-Cultural
SE	Scenario Events
SIGINT	Signals Intelligence
SIM	Social Impact Module
SME	Subject Matter Expert
STBL	Systems Technology Battle Laboratory
TEO	Task, Event, Outcome
TRAC-MTRY	TRADOC Analysis Center ” Monterey
TRAC-WSMR	TRADOC Analysis Center ” White Sands Missile Range
TRADOC	Training and Doctrine Command
TWG	Tactical Wargame
UAS	Unmanned Aerial Surveillance
UID	Unique Identification
XML	Extensible Markup Language

ACKNOWLEDGMENTS

I would like to thank the outstanding SIM programmers, Mr. Harold Yamauchi and Mr. John Ruck. Their tireless efforts over the past year resulted in quality documentation and model upgrades. Additionally, I would like to thank the Irregular Warfare Team at TRAC-WSMR. Ms. Kristen Clark, Mr. Benjamin Wintersteen, and Ms. Sarah Holden in particular provided valuable input into the development of the final versions of SIM. Finally, I'd like to thank LTC Jon Alt whose guidance kept the team on course throughout the project.

1. INTRODUCTION

1.1. OVERVIEW

This document provides an overview for the transition of the Social Impact Module (SIM). The transition occurred over a series of milestones than began with defining requirements and ended with the delivery of three software versions. TRADOC Analysis Center (TRAC) White Sands Missile Range (WSMR) TRAC-WSMR, the primary stakeholder, and TRAC-Monterey (MTRY), the model provider, worked in concert to ensure development adhered to written specifications.

1.1.1. End State

At the conclusion of SIM Transition, TRAC-WSMR owns SIM, a fully functional model that simulates population responses, key leader engagements, and social network activity. In addition, WSMR can employ the model for the Irregular Warfare (IW) Tactical Wargame (TWG) use case. After clearly defining a study question and conducting a measurement space workshop, WSMR will be able to design a scenario file for SIM that will provide outputs necessary to stimulate the players in future IW TWG.

1.1.2. Use Case

SIM is designed to support the following general use case:

- Human in the Loop (HITL) Simulation Exercise.
- Tactical level of war preferred; however, it is possible to aggregate data to simulate any population.
- Static and dynamic social networks represented in the location of interest.
- Server-based infrastructure modeling of services in the location of interest.

SIM is subject to the following assumptions:

- Survey data accessible for population in an IW TWG area of interest or sufficient population subject matter expertise to produce equivalent data.
- Information about threat, friendly, and neutral networks is known.
- Key leader information is available for individuals modeled in the networks.

When designing future IW TWGs, the study team first selects a study question and conducts a measurement space workshop. These meetings determine the requirements needed to construct a scenario file. The scenario file initializes SIM and outlines the population, key leaders, networks, events, infrastructure, and interactions used in the model. The requirements process will identify the data and methods necessary to build an appropriate scenario. The data most likely consists of surveys and subject matter expert (SME) input. The methods used to transform this data into scenario components will likely involve the application of data analysis techniques to survey data and involve the use of structured elicitation methods to gather SME input. Once complete, the scenario file defines the parameters required to produce outputs from SIM, and it determines how effective SIM is in meeting the requirement, stimulating a human player, for this use case. The ability of the IW TWG to answer the study question depends on multiple factors that influence human subjects, ultimately the focus of any wargame. SIM is just one of these factors. Determining the requirements early also provides the study team the time necessary to determine capability gaps in the wargame tools available.

This document identifies the transition process, discusses major decisions made to meet requirements, and provides an overview of the final capabilities of SIM. The design of this document is: first, an introduction; followed by the design of the SIM 1.0, 2.0 and 3.0, and finally, recommendations. The authors made a deliberate effort to keep the base document short with the most technical discussions in appendices.

1.2. BACKGROUND

1.2.1. Brief history of IW TWG

The purpose of TRAC's IW TWG is to investigate the potential effects of changes to Army doctrine, organization, or material in an IW environment. At the core of the wargame is the Social Impact Module. SIM adjudicates the effects of player actions on the local population in an area of interest and provides the Planning, Adjudication, and Visualization Environment (PAVE) tool with the output, current population opinions and issue stances. This output provides feedback to the human players participating in the wargame. PAVE is the players graphical user interface (GUI), and it is also the underlying database for all game information and activity. In this human subjects experiment, player actions provide stimulus. PAVE's ability to display changes in the game as a result of player actions is one of the most critical components of the wargame. SIM contributes to this stimulus by reporting responses from a population of interest. In addition to computer models, an Operational Wrap-Around (OWA) board game provides context and simulates the battalion level and above for the TWG.[3]

1.2.1.1. Proof of Principle

In 2009 the Proof of Principle (PoP) TWG showed that the Task, Event, Outcome (TEO) construct was feasible and the Cultural Geography (CG) model could replicate population opinions. Occurring in late-October 2009, the scenario mirrored current operations in Iraq;

however, the team declassified the final version of the game and all classified locations were removed. The original players were three platoon leaders, a company commander, and an insurgent commander. The Operational Wrap-Around provided a battalion commander and an additional insurgent commander.

1.2.1.2. Prototype 1.0

In 2010, the TWG scenario moved to Afghanistan. The team designed new TEOs and game mechanics improved. TRAC-MTRY added infrastructure to CG for modeling the effects of destroying and upgrading essential services. Dr. Deborah Duong, an Augustine Consulting Incorporated (ACI) contractor working with TRAC-MTRY, added a key leader social network called Nexus to the SIM. Nexus allowed a new type of intelligence: Critical Knowledge (CK). CK allowed players to discover new key leaders or infrastructure in their AOs.

The IW team changed the scenario to a brigade AO. As a result, PAVE required four company commanders and a battalion commander. The OWA provided their brigade commander. The team also added additional roles for host nation army and police in PAVE, and the OWA played the host nation government. The PAVE insurgent players were a Taliban and criminal faction. The OWA still contained only one Taliban commander. Finally, a Yellow player represented Non-Governmental Organizations (NGO) in both the OWA and PAVE.

Prototype 1.0 tested the ability of the TWG to support a study for the first time. The team executed the game in two (2) week-long sessions. The “Base Case” modeled operations in the AO from September 2009 until March 2010, before the Marjah offensive. The “Enabled Case” tested the effects of adding civil affairs teams to each company.

1.2.1.3. Prototype 2.0

In October 2011, the IW TWG team used the same Afghanistan scenario from Prototype 1.0, choosing to focus on streamlining game play and adding mechanics related to the study issue. This year, the TWG assessed the impacts of adding Company Intelligence Support Teams (CoISTs) to each company in the PAVE battalion. To meet this end, the team redesigned intelligence system to be more robust. Intelligence divided between human intelligence (HUMINT), signals intelligence (SIGINT), images intelligence (IMINT), and measurements and signals intelligence (MASINT). Additionally, game designers added a Prophet SIGINT collector and upgraded the representation of Unmanned Aerial Systems (UAS) to include the Shadow and Predator. Critical Knowledge had two characterizations: verified and unverified. Players could now verify their CK by exchanging it with other players or using other types of intelligence. The battalion commander also had an Intelligence Officer (S2 player) to help him with intelligence-related tasks. In the Enabled Case, four enlisted Soldier, CoIST specialists, worked with each company commander in PAVE to enhance operations.

The effort upgraded the model suite again. The models team added an infrastructure model, dubbed Infrastructure and Essential Services (IES), to the TWG tool set. This model, along with an economic model (IESE), not ready in time for wargame execution, would have enabled the second and third order effects of infrastructure sabotage and upgrades in CG.

1.2.2. After Action Review from Prototype IW TWG 2.0

The purpose of the TRAC-MTRY after action review (AAR) was to capture lessons learned from the FY11 IW TWG. The lessons learned informed the continued development of models and tools used for TWG support and highlighted issues for attention during the transition of SIM during FY12. Lessons learned from this AAR guided the stabilization effort for SIM 1.0, and SIM transition team ensured they considered each of the issues during subsequent development of SIM. From a holistic perspective, one of the key points from the AAR was the need to focus the wargame on the human subjects and to begin that process with a detailed measurement space workshop. See Appendix B for the complete AAR.

1.2.3. Need for transition

In 2011, the Director of TRAC issued guidance for TRAC-MTRY to transition the SIM to TRAC-WSMR for future use in IW TWG. Transitioning the capability provides TRAC-WSMR an in-house social simulation for the IW TWG use case. Equipping WSMR personnel with the knowledge and expertise to plan, develop, and execute SIM scenarios empowers their IW Team to conduct future TWG without the need for extensive model support from TRAC-MTRY.

1.3. OBJECTIVE

1.3.1. Problem statement and scope

1.3.1.1. Problem statement

The SIM Transition problem statement is to transition the SIM capability from TRAC-MTRY (model developer) to TRAC-WSMR (gaining organization) by 30 September 2012 in order to facilitate future iterations of the Irregular Warfare Tactical Wargame.

1.3.1.2. Scope

The team limits the scope of this project to the incremental improvement of SIM, training of WSMR personnel, and the documentation and delivery of the model. SIM transition is achieved through an iterative application of a systems design process to:

- Identify and clarify primary stakeholder needs and requirements.

- Identify potential approaches to address requirements.
- Propose and select solutions for implementation.
- Measure the effectiveness of the implemented solutions in meeting stakeholder requirements.

The development team and the gaining organization worked in close coordination throughout the process to identify requirements, evaluate the effectiveness of implemented solutions and to conduct training. Decisive to the success of the transition was the identification of gaining organization personnel to participate in training, requirements generation, and evaluation, in partnership with the development team

1.3.2. Constraints, limitations, assumptions

1.3.2.1. Constraints

- Models must be stable, simplified and integrated to finalize transition by 30 September 2012 (per TRAC Director guidance).
- TRAC-WSMR staff will only be available to support TRAC-MTRY on site transfer activities for a set period of time.

1.3.2.2. Limitations

- Data availability is based on use case scenario.
- Timelines may limit the ability to do extensive testing during implementation of key leader capabilities in the SIM.
- Only 1.5 contractor years funded for this project.
- TRAC-WSMR staff supporting transfer will simultaneously support all model integration activity.

1.3.2.3. Assumptions

- If limited data exists for scenario development, additional time will be available for data collection and SME elicitation.
- Leveraging student thesis work will support testing and evaluation plan.
- Contract support will be available to cover maintenance and model updates once SIM transfer is complete.
- TRAC-MTRY will provide technical model and wargame support after SIM transition.

1.4. DOCUMENT OVERVIEW

Chapter I introduces the problem, provides background and defines objectives for SIM Transition. Chapter II provides an overview of SIM 1.0. Chapter III details the addition of Key Leader Engagements and networks in SIM 2.0. It also describes the improvements made to the population model in SIM. Chapter IV explores the new modeling techniques employed by SIM 3.0, and explains the changes necessary to employ the model. Chapter V concludes the technical report and describes the handover of SIM to TRAC-WSMR. A series of appendices explain technical details including testing of the model.

2. SIM 1.0

This chapter will provide an overview of SIM 1.0, the baseline module used in support of the IW TWG11. After the IW TWG11, TRAC-MTRY conducted a thorough after action review (AAR). Immediately following the AAR, work began on the stabilization of the models in SIM. The transition team emphasized the importance of quality documentation to accompany the software delivered to WSMR. The team initially focused on updating the JavaDocs and User Guide. Finally, rigorous testing ensured the model produced traceable and explainable results. After SIM 1.0 stabilization was complete, TRAC-WSMR analyst Ms. Kristen Clark served as a visiting analyst from late-February until late-April 2012 in order to receive training on the proper use of SIM 1.0. This chapter describes the stabilized version.

2.1. SIM 1.0 BASICS

This section provides background information on SIM 1.0 and its components.

2.1.1. Population Model

The Social Impact Module version 1.0 is comprised of the Cultural Geography (CG) model and Nexus Network Learner (NNL) models. CG is an agent-based model of the operational environment based on doctrine and social theory. CG consists of agents (simulated people) interacting with each other in the conflict environment and responding to wargame player actions within the environment. Each agent is defined by a set of demographic dimensions that collectively shape the agent's beliefs, values, interests, stances on issues, and behaviors. The narrative paradigm is the underlying social theory upon which narrative identities are developed from data to form agent beliefs, values, and interests.[2] The narrative paradigm primarily manifests itself in the data (scenario) development process; however, it also forms the foundation for how an agent perceives events in the simulation.

Leveraging survey data, SIM 1.0 models a population's beliefs, values, and interests (BVI). The SIM team partitions the population according to how counterinsurgents must understand the environment in FM 3-24 (p 1-22, para. 1-124)[4]:

- Organization of key groups in the society.
- Relationships and tensions among groups.
- Ideologies and narratives that resonate with groups.
- Values of groups, interests, and motivations.
- Means by which groups communicate.

- The society's leadership system.

Once partitioned, scenario developers identify beliefs from the available population data. Those beliefs map to issue stances using a Bayesian Belief Network (BBN). Scenario designers initialize these beliefs and issue stances from survey data. SIM updates these beliefs over time as part of model execution. The BBN relate core beliefs to both population perceptions and to observed attitudes and behaviors (OAB) toward various factions in the game. For example, a population agent has an OAB toward coalition forces and a separate OAB toward their government. Population agents have an OAB for each actor in the TWG. Within SIM 1.0, scenario events (SE) trigger updates to the BBN for both OAB and issue stances. Appendix C discusses how the model executes this process of updating agent beliefs.

2.1.2. Key Leader Engagements

The key leader component in SIM enables TWG players to meet with key leaders and conduct simulated key leader engagements (KLE). KLE can result in wargame participants gaining useful information such as knowledge about key leaders, threat networks, or general knowledge contained in the TWG database. For SIM 1.0, Nexus Network Learner, or simply Nexus, is the model that handles those engagements.

The key leader model also contains static and dynamic social networks within the simulation. These networks create knowledge that players may discover during the course of the game. Much of the knowledge is noise from behaviors of the agents; however, some of this knowledge is critical knowledge about a variety of subjects including threat networks and enemy activity. These networks model the social, professional, personal, criminal, and threat networks that exist within a population. Networks in the model determine who agents communicate with and the range of possible outcomes.

2.2. SIM 1.0 MODEL INTEGRATION

2.2.1. Interaction with PAVE

In the SIM scenario file, there is a PAVE interface scenario worksheet named "PaveInterface". In the worksheet, a scenario developer specifies a PAVE database by name and location (path). For the TWG use case, both CG and Nexus utilize a "warm-up" period. This warm-up period conditions the population agents in the model. This is not required; however, it is a recommended best practice. The warm-up period provides population agents with evidence of events that may happen later in the game. This evidence establishes an agent's "memory" and enables believable reactions based on subject matter expert input to the scenario file. Without a warm-up period, population agents exhibit drastic shifts in opinions. Once enough evidence accumulates, the virtual population in SIM demonstrates stable behavior. If the wargame is looking at a population's possible reaction to invasion, a much smaller warm-up period might be appropriate. The final result is a model that simulates reasonable population responses to human-player actions in a wargame.

After the warm-up period, the wargame begins and players can input their actions into PAVE. Upon completion of the planning phase, Nexus runs to determine the results of planned actions with agents in the key leader model. When a Nexus run finishes, some of the possible outcomes are events scheduled in CG via the PAVE database. Next, CG runs to determine population agent responses to game events. CG writes the results back to the appropriate PAVE database tables, and the PAVE GUI displays the results back to players. It is worth noting that there are pitfalls with aggregating population agent opinions by location in PAVE. SIM calculates opinions for each agent; however, those opinions are displayed as a single opinion per location to the players. If there are several agents in a location (hex), many agent opinions can be obscured, depending on the algorithm used to calculate the overall classification of the hex.

2.2.2. Integration Testing

The initial testing of SIM 1.0 employed a top-down approach. After TRAC-MTRY's IW TWG 2011 AAR, the team focused on determining why it was difficult to discern significant results coming from the model. The initial hypothesis attributed the behavior to the representation of the complex conflict ecosystem. Actions taken by blue players, tactics employed by the enemy (red) player, infrastructure needs, and communication by agents created a significant amount of "noise" that was not intentionally isolated during the last TWG. Lead by TRAC-MTRY analyst MAJ Paul Evangelista, the transition team decided to peel away these layers one-by-one during testing.

First, the team executed the IW TWG 2011 model runs again and examined the results. MAJ Evangelista determined that there were too many independent variables to isolate causality for why agents did not show significant increases or decreases in issue stances and observed attitudes and behaviors (OAB). These were important discoveries and provide evidence for simplifying scenario complexity in the future. Specifically, a few of the recommendations were:

- Fewer stereotypes or increase the amount of data collected for each stereotype.
- Develop fewer scenario events or bin the events.
- Provide consistent enemy actions for comparison of TWG cases.

Based on this analysis, the team began stripping away complexity in the IW TWG 2011 scenario file and conducted thread testing to determine the variables that produce significant results from the model. The initial runs involved SIM alone; however, the team eventually ran the same scenario with PAVE and compared these iterations. The comparison confirmed that the exact same output from the model was written to the PAVE database.

2.3. SIM 1.0 DESIGN

2.3.1. Scenario Files and Data

In NPS MOVES Institute faculty member Dr. Buss's Discrete Event Simulation (DES) Modeling guide, he states that, "the primary purpose of a DES simulation model is to learn something about the system being modeled that wasn't known previously." He continues by highlighting the importance of creating, "DES models with an eye towards the analysis that will ultimately be performed on them."[1] Dr. Buss's comments highlight the importance of fully understanding the specific use cases for the models and the analysis that will be conducted on model output. Understanding the inputs and outputs necessary to answer a study question is critical for the successful employment of SIM and will vary, depending on the wargame location, study question, and measurement space. Data needed by the analysts determine the type and number of loggers required in the code. Loggers track changes to state variables in the model due to agent activity. Because SIM is a discrete event simulation, state variables can be known at any given time; however, tracking every variable in SIM will result in an overwhelming amount of data. It would also degrade the performance of the model. A measured, deliberate approach to the data required has the highest probability of producing the necessary outputs from SIM while simultaneously improving model performance.

2.3.1.1. Inputs

The population model in SIM is very flexible. A Microsoft Excel scenario file initializes the model and establishes the framework for model outcomes. Discussing the complete scenario file is beyond the scope of this report; however, Appendix H contains the SIM User Guide and explains in great detail each variable required to build a SIM scenario. This section will highlight some of the critical components in a SIM scenario file.

Most importantly, SIM requires agent prototypes that define the attributes for a population agent. Agent prototypes reference specific case files. The design team develops agent case files from survey data and subject matter expert elicitation using a variety of methods outlined in Appendix H and Appendix J of this report. It is impossible to cover an exhaustive list of variables contained in the case files and scenario file because the scenario developer can use whatever actors, population stereotypes, beliefs, issues, attitudes, actions, infrastructure or effects deemed necessary to accomplish the objectives of the wargame.

The SIM scenario file must contain the list of possible events to process from the wargame. These events should have an associated effect on each of the agents. The scenario design does not need to map an effect from every event to each agent prototype; however, if there is not an effect mapped in the scenario file, then that event will have no effect on the neglected population agent. SME elicitation largely determines the effects that an event will have on each agent prototype. For example, if there was an event called “Coalition Forces conduct a security patrol”, there would be some expected effect on the population. Assume that effect had a 70-percent chance of being perceived as positive event on the agent prototype “Christian Males Over-30”. Thus, if a HITL coalition force player conducts a security patrol in the TWG, there is a 70-percent chance that a positive response would result in population agents in SIM characterized as Christian males over-30 years of age who witnessed the event. Note that in practice, from the standpoint of SIM, a finite number of SIM events could be mapped to any number of player events and likely achieve more informative results than in previous games where the mapping was one-to-one. The larger number of events only served to increase noise within the analysis since many player events were equivalent from the perspective of the agent.

SIM represents infrastructure with multi-server queues. These servers are defined in the scenario file. This can mirror an actual area of operations, or it can be general, if infrastructure is not a critical component of the game. Scenario developers define an agent’s infrastructure needs in the scenario file. Agents seek infrastructure throughout SIM execution automatically. The scenario file also specifies who “owns” each infrastructure server. This determines how an agent reacts when their needs are not met. Much like events, infrastructure effects determine how a population agent reacts to receiving or failing to receive their infrastructure needs. Note that other infrastructure models are at play within the wargame construct, but this effort does not address their integration within SIM. It is not yet clear what value these alternative models provide the IW TWG construct, but it is clear that their integration is challenging.

2.3.1.2. Outputs

After a model run, SIM outputs a series of comma separated value (.csv) files. The loggers in the model determine the data written to the output files. SIM copies the logged data to files defined in the scenario file. As with the inputs, there are not a set number of output files, it depends largely on the number of issues and actors in the wargame. Each issue has an associated file for tracking population issue stances. Each actor also has an associated output file for recording the OAB changes pertaining to that actor. During a wargame, SIM also writes the data specified by the IW TWG team to the PAVE database.

Note that OABs are the primary stimulus provided to the player during the game through PAVE . If not designed and tested prior to a wargame, the OABs output from SIM could be very confusing. The current method of binning OABs into one of five categories was not intuitive, and thus SIM 2.0 and SIM 3.0 explored alternate methods of implementation, discussed later in this document.

2.3.2. Structural overview of SIM 1.0.

Implemented using SimKit Libraries for Java, SIM leverages the component structure of SimKit for an object-oriented model design. For example, SIM 1.0 contains agent, perception, attention, memory, and action selection components. These are not the only modules in SIM, but they are some of the most important. One other important facet of SIM is communication. Communication occurs between the agents in the model based on homophily and propinquity. Homophily is a social distance calculation done for each pair of agents based on demographic dimensions and issues stances, specified in the scenario file. Homophily is a number between zero (0) and one (1) where zero represents two agents who are nothing alike and one is a pair of agents who are exactly alike. The process of building a homophily network expressed in three equations:

Difference in social dimension (Δd)

$$kd_{ij} = i_d - j_d \tag{2.1}$$

where i_d is the position occupied by agent i in dimension d and j_d is the position occupied by agent j in dimension d.

Social Distance:

$$s_{ij} = \sqrt{c_1\left(\frac{k_{1_{ij}}}{\max k_1}\right)^2 + c_2\left(\frac{k_{2_{ij}}}{\max k_2}\right)^2 + \cdots + c_n\left(\frac{k_{n_{ij}}}{\max k_n}\right)^2} \tag{2.2}$$

where s_{ij} is the Eucledian distance between agent i and agent j across all n dimensions, and c is an optional coefficient.

Link Weight:

$$w_{ij} = 1 - \left(\frac{s_{ij}}{\sqrt{d}}\right) \tag{2.3}$$

where w_{ij} is the link weight between agents and d is the number of dimensions present.

Every agent in the social network is linked with every other agent with link weights (w) between 0 and 1. Minimum thresholds in the scenario file determine what link weight an agent must have to communicate with another agent.

Propinquity is a straight line calculation of physical (map) distance. The scenario file contains a minimum distance threshold for communication between agents.

Generally speaking, the scenario file specifies the number and type of agents in the model. SIM instantiates population agents using the agent constructor in the Java source code. Agents are SimEntities, as defined by the SimKit Java Libraries. An agent perceives events and seeks infrastructure in the model. Often, agents have the choice between many percepts to process. Based on their memory and a calculated utility, an agent makes a choice about what action to take or event to process. These choices trigger calculations about their issue stances or OABs. An agent can communicate about events, seek resources, or do nothing. This basic procedure repeats itself continuously.

See Appendix C for a technical overview of these key SIM components. For an in depth understanding of SIM components, refer to the JavaDocs and Dr. Buss's Manual on Discrete Event Simulation Modeling.

2.4. SIM 1.0 TESTING

2.4.1. Objective

SIM 1.0 testing verified that the model works correctly and is stable. Integration testing ensured that SIM was functionally interoperable with PAVE and produced outputs necessary for conducting future IW TWG. See Appendix D for results and analysis of SIM 1.0 Testing.

2.4.2. Overview

The testing of SIM 1.0 began in January 2012. In an attempt to save time, the team decided to use the data and scenario files from the TWG11.Still taking a top-down approach, the team modified the TWG11 scenario file by reducing the data to just a few population agents, events, and infrastructure nodes. There were flaws with this method so a refined list of stereotypes were developed. Finally, there were behaviors that were difficult to isolate because the scenario contained multiple stereotypes so a robust set of single-agent scenarios were created to show the range of possible values that could be produced from the model, given strictly controlled inputs.

The remainder of this section covers the testing done on the population model in SIM 1.0. See Appendix D for more analysis of the resulting data. A complete Digital Versatile Disc (DVD) of testing data preceded this report to TRAC-WSMR, delivered on 21 September 2012. The DVD contained over 200 scenario files employed during SIM 1.0 testing and over 1400 output files. The TRAC-MTRY team analyzed these output files for significant findings; however, TRAC-WSMR could easily modify the input files, execute the scenario in SIM, and relook at the data. Providing these files can greatly reduce the time to conduct follow-on testing for analysis.

2.4.3. Iterations

2.4.3.1. First Iteration: Training Phase

The first scenarios tested was a 5-agent scenario derived from the TWG11 scenario. At the time, TRAC-MTRY analyst MAJ Brown was teaching LTC Caldwell how to develop scenarios for SIM. Learning to properly develop a scenario file took approximately one week and teaching scenario development using a existing scenario and walking through each of the 60-worksheets was easiest. The team developed three scenario files and analyzed the resulting data.

2.4.3.1.1. Settings

The discount factor, lambda (λ) affects an agent's "memory" in the cognitive architecture. It is a measure of how fast they discount, or forget, events occurring in the past. This variable has a significant affect in the model, discussed in detail throughout Appendix D. Time units in SIM are notional. For the TWG, time is measured in days and weeks, so the remainder of this document will use that construct. General settings remained the same as in the IW TWG11 scenario file:

- Case Files: TWG11 Case Files for specific agents.
- Discount Factor Lambda (λ): 0.5
- Single location (hex).
- 168 time units (24-weeks).

2.4.3.1.2. Agents

The three scenarios contained five agents. The team chose to use five agents in order to create the conditions for communication between like agents. Five agents enabled two pairings of like agents and a single agent without a stereotype pairing for social distance similarity (social distance or homophily). These first tests used the following stereotypes from IW TWG11:

$A\_P\_R\_F\_Ma$	An achieved, pro-government, rural, fundamentalist, military-aged male.
$I\_Pa\_U\_M\_Ma$	An inherited, passive, urban, modernist, military-aged male.
$Un\_V\_R\_S\_Sp$	An unemployed, violent, rural, secular, Spin Giri.

In this scenario, there were $2 \times A\_P\_R\_F\_Ma$, $2 \times Un\_V\_R\_S\_SP$, and $1 \times I\_Pa\_U\_M\_Ma$. The decision to use this mix of stereotypes was based entirely on MAJ Brown's recent experience with the scenario files. He developed the CG scenario file for use in the IW TWG 2011. The stereotypes were chosen based on what their demographic dimensions represented and not the quality of these particular population agents. The team identified assumptions about the underlying case files as a problem later in the testing process.

2.4.3.1.3. Events

The first "training" scenario (TS-1) developed by the team used the event list from the TWG11 Base Case. The TWG11 events associated with the three stereotypes in this scenario were used, and all other events were discarded. After eliminating TWG11 events that did not pertain to the five agents in this scenario, 10,578 events remained in the file. All five agents were in the same hexagon (hex), and the team included each type of infrastructure node used in the TWG11 to ensure the agent was able to satisfy his basic needs. The hex's infrastructure included:

- Electricity
- Employment
- Farm Supplies
- Legal
- Medical

After analyzing the resulting output files, it was obvious that the use of several stereotypes,events, and infrastructure confounded the results. It was very difficult to attribute outputs to the scenario input file without including several data loggers. The team decided to shelve events and focus on infrastructure for the next set of scenarios.

2.4.3.1.4. Infrastructure

The second "training" scenario (TS-2) developed by MAJ Brown to train LTC Caldwell isolated the infrastructure nodes in location AC163, an arbitrary single hex chosen for testing. This decision allowed the research team to know exactly where agents would go to receive their infrastructure needs. The scenario employed the following infrastructure nodes from the TWG11 Base Case files:

- Electricity
- Employment
- Employment_Nontechnical_Elec_Dist
- Employment_Nontechnical_Elec_Gen
- Employment_Nontechnical_FarmSupplies
- Employment_Nontechnical_Medical
- Employment_Technical_Elec_Dist
- Employment_Technical_Elec_Gen
- Employment_Technical_Medical
- FarmSupplies

- Legal_Red_Mobile
- MEDCAPSupplies
- Medical
- Medical_Red_Mobile

After running this scenario and analyzing the output files, the team decided to further isolate the infrastructure nodes (servers). To do this, the team developed a third "training" scenario (TS-3). TS-3 removed infrastructure complexity from TS-2. The only infrastructure node in TS-3 was a single electricity node. The use of one electricity node allowed the team to isolate agent activity. The resulting output provided a better understanding of agent actions, issue stances, and attitudes. TS-3 represented the first time it was relatively easy to discern the effects of single inputs on the outputs from SIM. The team determined that this would be an acceptable method for testing inputs and outputs as the testing plan progressed.

2.4.3.2. Second Iteration: Refining the testing strategy

After TRAC-MTRY internal training concluded, the team refined the test strategy. The first iteration of testing revealed results similar to IW TWG11 outputs. Due to the complex environment modeled in SIM, numerous events and infrastructure created several possible choices for the population agents in the model. This resulted in both positive and negative outcomes. In the short run, those results created difficulties for analysts trying to determine the causes for changes in issue stances or OAB. The training scenarios revealed a need to reduce the complexity even more.

2.4.3.2.1. Settings

There were no significant changes to the general settings used in the second wave of tests. Applying the lessons learned from the three training scenarios, the team chose to eliminate additional variability by using only two agent prototypes, a single event, and a single infrastructure node. By limiting the factors affecting agents in the model, the team sought to create traceable results in SIM output files. Table 2.1 outlines the experimental design for the second iteration of tests. The general settings remained the same as the first iteration:

- Case Files: TWG11 Case Files for specific agents.
- Discount Factor Lambda (λ): 0.5.
- Single location (hex) used.
- 168 time units (24 weeks).

Design Point	Agents	Stereotypes	Location	Scripted Actions	Sociability Method	Socialbility Class Distrib.	Infrastructure
1	1	I_P_U_S_Sp	AC163	CFOperatesInArea	Not applicable	Not applicable	None
2	2	I_P_U_S_Sp	AC163	CFOperatesInArea	K_NEAREST_NEIGHBOR	exponential(2.5)	None
3	2	I_P_U_S_Sp, Un_V_R_F_Ma	AC163	CFOperatesInArea	K_TRIM_THRESHOLD	constant(0.99)	None
4	1	I_P_U_S_Sp	AC163	None	Not applicable	Not applicable	InfraElectricity
5	2	I_P_U_S_Sp	AC163	None	K_NEAREST_NEIGHBOR	exponential(2.5)	InfraElectricity
6	2	I_P_U_S_Sp, Un_V_R_F_Ma	AC163	None	K_TRIM_THRESHOLD	constant(0.99)	InfraElectricity
7	1	I_P_U_S_Sp	AC163	CFOperatesInArea	Not applicable	Not applicable	InfraElectricity
8	2	I_P_U_S_Sp	AC163	CFOperatesInArea	K_NEAREST_NEIGHBOR	exponential(2.5)	InfraElectricity
9	2	I_P_U_S_Sp, Un_V_R_F_Ma	AC163	CFOperatesInArea	K_TRIM_THRESHOLD	constant(0.99)	InfraElectricity

Table 2.1: Design Points for second iteration of test cases.

2.4.3.2.2. Agents

The second set of tests used the following stereotypes from TWG11:

- I_P_U_S_Sp. An inherited, pro-government, urban, secular, Spin Ghri.
- Un_V_R_F_Ma. An unemployed, violent, rural, fundamentalist, military-aged male.

The team chose these stereotypes because the narratives represented a mix of very different population demographic dimensions. The intent was to get a mix of agents from one stereotype that would not communicate with the other stereotype due to low link weights (homophily).

2.4.3.2.3. Events

Unlike the first iteration that used all the events from the IW TWG11 events list, the team scripted only one action per day. That single event was a coalition patrol in the hex occupied by the agents. By using one event, the team ensured that an event's effects were understood. This was a good first step for verifying model results by limiting the scenario file inputs.

2.4.3.2.4. Infrastructure

The SIM transition team limited the number and types of infrastructure servers in the same way they limited events. Developers crafted scenario files that constrained an agent's needs to one infrastructure type. In this case, the infrastructure was electricity. Then, the team added the electricity infrastructure server to the scenario file in the hex containing the population agents. This decision ensured that agents properly sought electricity, and it verified that the server construct worked properly for satisfying infrastructure needs.

2.4.3.3. Third Iteration: Fixing communications

During the second set of tests, the team identified a potential issue with agent communication. This issue involved the model's method for determining how the cognitive architecture handles percepts. In short, percepts are Java objects arriving in discrete intervals. An agent is "aware" of every percept. The TWG11 implementation of communication allowed an agent to handle a specified amount of percepts at one time, known as a "situation". The agent chose the most relevant percept from the situation to act upon. The remaining percepts were discarded. Discussion of this issue by the Social-Cultural Methods, models, and Analysis Working Group (MmAWG) during a visit to TRAC-MTRY, convinced the model development team that this was not the best implementation. This method did not enable agents to process multiple, relevant issues. Therefore, developers implemented a modification that allowed recycling communication percepts back into the queue for processing by the population model. This change necessitated re-running the design points from the second set of tests, and all future testing contained this improvement.

2.4.3.4. Fourth Iteration: Useful Results

After noticing anomalous behavior with the stereotypes in the second and third iteration of testing, the team decided to look closer at the stereotype case files. It was very apparent to the analysts that a lack of survey respondents affected the output from the model. A lack of data supporting the underlying BBN caused this behavior. If there is no evidence for a particular combination of beliefs, the Bayesian solution assigns it a 50-50 possibility " a coin flip. This leads to confusing model outputs because a population agent reacts to events with the same 50-50 probability of reacting positively or negatively. The resulting effect is SIM outputs converging to 50-percent on a 0 to 100-percent scale.

Scenario developers should avoid choosing population samples that lack evidence. Figure 2.1 shows the population agents in the TWG11. It is evident that very few population stereotypes contained the evidence necessary to properly populate the BBN. Each population stereotype should have a sampling of people that either avoids the coin-flip necessary to fill out the BBN or overwhelms it with evidence. The transition team tested these factors extensively in follow-on tests. Using a smaller number of population stereotypes that contain a larger number of survey respondents per stereotype is recommended for future IW TWG.

2.4.3.4.1. Settings

This iteration of testing utilized a new set of agents. Applying the lessons learned from previous testing iterations, the team chose to select agents based on the number of survey respondents contained in the agent prototype case file. By ensuring a larger representation of survey respondents in the underlying files, the team sought to eliminate the "by-chance" results in SIM output files. Table 2.2 outlines the experimental design for the fourth iteration of tests. The general settings remained the same as previous iterations:

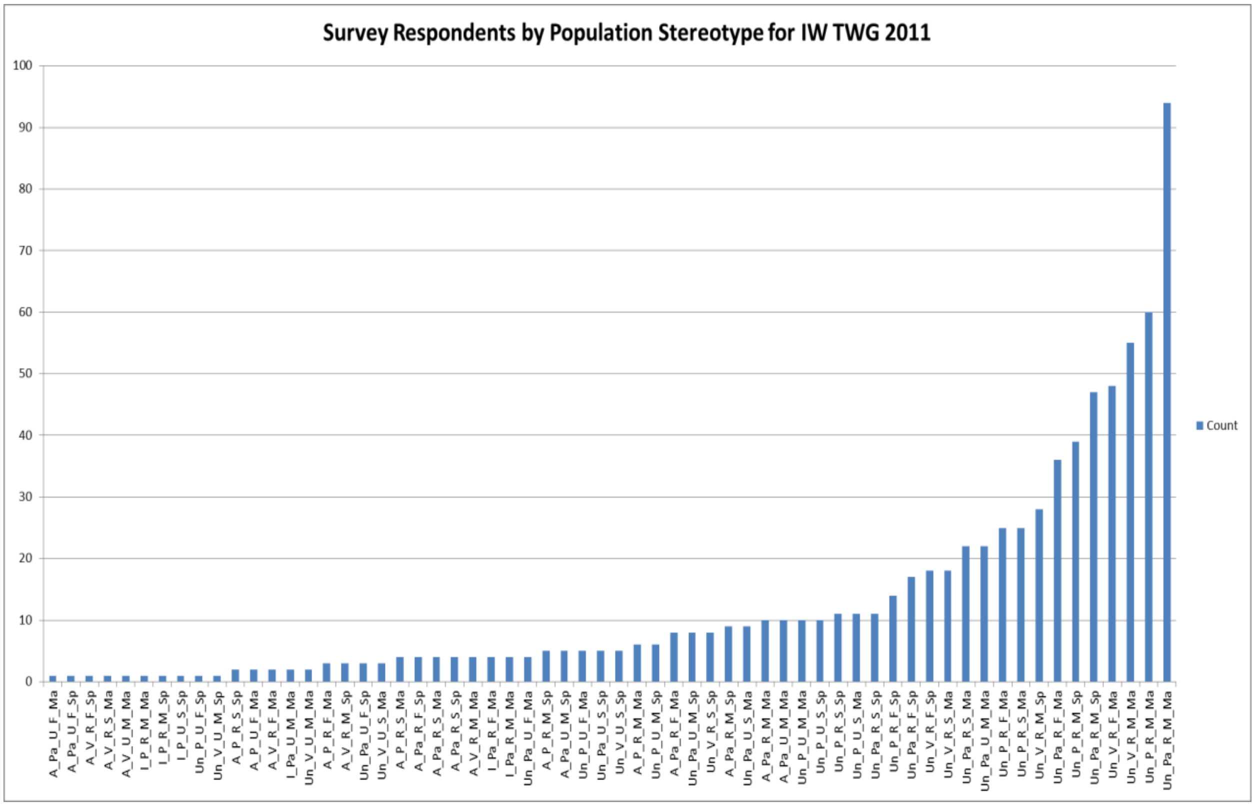

Figure 2.1: IW TWG11 Population Stereotype Counts

- Case Files: TWG11 Case Files for specific agents.
- Discount Factor Lambda (λ): 0.5.
- Single location (hex) used.
- 168 time units (24 weeks).

2.4.3.4.2. Agents

The fourth iterations of tests used the following stereotypes from TWG11:

- Un_Pa_R_M_Ma: Unemployed, Passive, Rural, Moderate, Military-aged male.
- Un_V_R_F_Ma: Unemployed, Violent, Rural, Fundamentalist, Military-aged male.

The team chose these stereotypes because the case files were more mature than other stereotypes. Un_Pa_R_M_Ma had 94 survey respondents, while Un_V_R_F_Ma had 48. Although these agent prototypes were similar in three demographic dimensions, the narratives represented a mix of very different population agents ” passive-moderates vs. violent-fundamentalists.

2.4.3.4.3. Events

The team continued to use only one scripted action per day. As in the previous two iterations, that single event was a coalition patrol in the hex occupied by the agents. The team repeated this technique because the effects were isolated and understood.

2.4.3.4.4. Infrastructure

The SIM transition team also continued to limit the number and types of infrastructure servers. Agent's needs remained constrained to one infrastructure type (electricity). The electricity infrastructure server was in the hex containing the population agents ensuring that agents properly sought electricity. The electricity infrastructure server was in the hex containing the population agents, ensuring that agents properly sought electricity.

Design Point	Agents	Stereotypes	Location	Scripted Actions	Sociability Method	Socialbility Class Distrib.	Infrastructure
10	1	Un_Pa_R_M_Ma	AC163	CFOperatesInArea	Not applicable	Not applicable	None
11	2	Un_Pa_R_M_Ma	AC163	CFOperatesInArea	K_NEAREST_NEIGHBOR	exponential(2.5)	None
12	2	Un_Pa_R_M_Ma, Un_V_R_F_Ma	AC163	CFOperatesInArea	K_TRIM_THRESHOLD	constant(0.99)	None
13	1	Un_Pa_R_M_Ma	AC163	None	Not applicable	Not applicable	InfraElectricity
14	2	Un_Pa_R_M_Ma	AC163	None	K_NEAREST_NEIGHBOR	exponential(2.5)	InfraElectricity
15	2	Un_Pa_R_M_Ma, Un_V_R_F_Ma	AC163	None	K_TRIM_THRESHOLD	constant(0.99)	InfraElectricity
16	1	Un_Pa_R_M_Ma	AC163	CFOperatesInArea	Not applicable	Not applicable	InfraElectricity
17	2	Un_Pa_R_M_Ma	AC163	CFOperatesInArea	K_NEAREST_NEIGHBOR	exponential(2.5)	InfraElectricity
18	2	Un_Pa_R_M_Ma, Un_V_R_F_Ma	AC163	CFOperatesInArea	K_TRIM_THRESHOLD	constant(0.99)	InfraElectricity

Table 2.2: Design Points for fourth set of tests.

2.4.3.5. Fifth Iteration: Establishing a baseline

After Ms. Kristen Clark departed TRAC-MTRY at the conclusion of her training, it was time to combine many of the lessons learned and examine results from the model using extreme values. This type of testing verified that the model produced expected results, and it demonstrated the range of possible results. A very deliberate design of experiment (DoE) isolated a single variable at a time to allow pair-wise comparisons and Analysis of Variance (ANOVA) tests. This change marked the beginning of the bottom-up testing approach whereby the team established a baseline and built in additional complexity, using the baseline as a foundation for follow-on tests.

2.4.3.5.1. Settings

This iteration of testing utilized a single agent whose case file was created to contain specific beliefs and issue stances. This technique allowed the research team to determine the effects of case file survey responses on the model. Previously, SIM had been tested with actual data and using Java JUnit tests; however, this was the first time the team tested the model with a controlled set of survey responses. The team was able to isolate the effects of survey respondent numbers, discount factors, agent beliefs, scripted event effects, and infrastructure availability. This testing took months to accomplish; however, the results were invaluable. General settings for this iteration of testing were:

- Case Files: Design team crafted case files for controlled responses.
- Discount Factor Lambda (λ): $\{0.99, 0.5, 0.1, 0.01, 0.001\}$.
- Single location (hex) used.
- 168 time units (24 weeks).

2.4.3.5.2. Agents

The fifth set of tests used the stereotype name, I_P_U_S_SP; however, the testing team developed a specific set of case files for testing that controlled the responses. The purpose was to evaluate the effect that the number of responses has the rate of change in the model. Table 2.3 outlines the variations of agent responses tested in this iteration of testing:

2.4.3.5.3. Events

The team limited events in this iteration of testing to a single event occurring per day. That event was a coalition patrol (CFOperatesInArea) that occurred in the hex containing the agent. The effects of the patrol were varied on extremes in the same manner as agent responses. The population agents either viewed events as 100% positive or 100% negative, and the team evaluated all combinations. Table 2.4 provides an overview of the first 56 design points of this iteration. The highlighted columns illustrate variable changes and the focus of analysis for that DP. For example, DP_1 and DP_2 differ by the belief-issue stance combination, and DP_1 through DP_7 differ from DP_8 through DP_14 by a positive vs. negative effect of the coalition force patrol.

2.4.3.5.4. Infrastructure

The research team did not utilize infrastructure nodes in this iteration of testing. Instead, the team attempted to isolate the effects of scripted events on the population. By not adding the complexity of infrastructure needs, noise from communications also disappeared, making the output data easier to analyze.

2.4.4. Significant Findings

SIM 1.0 testing focused on ensuring that model outputs were traceable and explainable. During the test and evaluation process, a testing strategy materialized for testing future versions of SIM. Limiting the scenario files to a single agent, or a few agents, experiencing a limited number of events produced the type of output data necessary to accomplish the

Number of Respondents	Belief		Issue Stance		OAB	
	Adequate	Inadequate	Adequate	Inadequate	Negative Active	Positive Active
100	1	99	1	99	0.99	0.01
	99	1	99	1	0.01	0.99
	50	50	50	50	0.5	0.5
	50	50	50	50	0.99	0.01
	50	50	50	50	0.01	0.22
	100	0	100	0	0.01	0.99
	0	100	0	100	0.99	0.01
1000	1	999	1	999	0.99	0.01
	999	1	999	1	0.01	0.99
	500	500	500	500	0.05	0.05
	500	500	500	500	0.99	0.01
	500	500	500	500	0.01	0.99
	1000	0	1000	0	0.01	0.99
	0	1000	0	1000	0.99	0.01
10	1	9	1	9	0.99	0.01
	9	1	9	1	0.01	0.99
	5	5	5	5	0.5	0.5
	5	5	5	5	0.99	0.01
	5	5	5	5	0.01	0.99
	10	0	10	0	0.01	0.99
	0	10	0	10	0.99	0.01

Table 2.3: Fifth Iteration of Testing: Case file combinations.

objectives. The test output showed extremely predictable results from controlled inputs. Appendix D contains descriptions of the most significant findings to include:

- Result differences are not statistical significance with more than 100 survey respondents per stereotype; however, there is statistically significance differences when using fewer survey responses such as 10 per stereotype.
- The discount factor lambda (λ) has a significant result on outputs from the model.
- Agent issue stances and OABs are asymptotic depending on the configuration of the model due to the BBN implementation.

2.5. SIM 1.0 TRAINING

2.5.1. Overview

Training for SIM 1.0 occurred between 27 Feb 2012 and 27 Apr 2012. This period of time coincided with Ms. Kristen Clark's rotation as a visiting analyst at TRAC-MTRY. The IW Lead designated Ms. Clark as the TRAC-WSMR SIM Point of Contact (POC). She was on site for the initial visit of the SC MmAWG from12-15 March 2012. The MmAWG visited to examine the socio-cultural underpinnings of SIM. This time benefited both the MmAWG and Ms. Clark by allocating a week to examine the social theories behind SIM. Deliberate effort was spent to develop products for the MmAWG that clearly outlined and demonstrated SIM capabilities and limitations. Ms. Clark was involved in developing these briefings and tools, deepening her understanding of the model The primary effort during Ms. Clark's time with TRAC-MTRY was learning how to build a scenario file. This took a considerable amount of time; however, by the end of her training, she produced a scenario file from scratch that the team used to conduct integration testing with PAVE in the Defense Information Systems Agency (DISA) Rapid Access Computing Environment (RACE) online test environment. Upon her return to TRAC-WSMR, Ms. Clark conducted a Professional Development (PD) workshop with the IW team. This workshop covered the basics of developing a SIM scenario file and the lessons learned from her time with TRAC-MTRY.

2.5.2. Objectives

The team scheduled specific objectives in three phases. This section describes Phase I and II accomplished onsite during the initial training period. MAJ Richard Brown led Phase I tasks, and LTC Jason Caldwell directed Phase II.

2.5.2.1. Phase I: TWG Specific Scenario Development Tasks

- Understand Theoretical Underpinnings.
  - Know the social theory of the model and how it ties to the conceptual model.
  - Understand what portions of human behavior are being explained by various social theories.
- Population Data Development.
  - Partition populations based on survey results and supporting data through factor analysis techniques (identify demographic and build stereotypes).
  - Build population narratives that inform the population partitioning, by population demographic.
  - Develop population beliefs, values and interest (per FM 3-24.2 COIN doctrine) that will change as agents are stimulated by events.

  - Identify issues important to the use case population - issue stances.
  - Map and transform survey results (specific questions) to beliefs, issues, and OAB states to initials agents.
  - Construct initialization files (case files) based on survey data that represent the population partitioning.
- Bayesian Belief Networks.
  - Construct Bayesian Belief Networks that generate state changes in the agents and demonstrate an agent's position on an issue or its OAB for a given actor in the wargame.
  - Instantiate all BBNs for issues and OAB with set of unique stereotypes.
  - Understand and can explain the calculations occurring with the BBN and can identify limitations with respect to IW TWG use case.
- Scenario Event Development.
  - Develop appropriate scenario events that represent the outcomes generated from the task-event-outcome framework in TWG.
  - Develop the survey instrument to elicit from a set of SME the response, by stereotype, to the total set of scenario events that can be run in the game.
- Represent the set of SME responses to each scenario event as a distribution from which SIM draws the effects on beliefs for each issue and OAB state change.
- Scenario File Generation.
  - Understand how to construct and manipulate a base TWG SIM scenario file.
  - Understand how to populate the SIM scenario file with data from the population scenario data development process.
- Running the Model.
  - Understand how to install and run software, to include preparation of environment variables and all necessary additional software.

2.5.2.2. Phase II: Data Development and Analysis

- Learn Key Leader Instantiation.
  - Develop key leader network per TWG key leader representation requirements.
  - Instantiate key leaders within SIM framework per the SIM Key Leader representation capability.
  - Refine capability/representation as necessary for TWG.

- Data Source Development.
  - Identify Data Sources needed to construct a population behavior scenario.
  - Identify and begin relationship with SME to shape data source identification for next IW TWG.
- Social Network Analysis.
  - Develop the underpinnings for social distance calculations in the model based on use case population demographic dimensions.
  - Iterate social distances with identified population SME.
  - Understand high level methods for examining the population social network as appropriate for TWG use case requirements.
- Infrastructure and Essential Services.
  - Identify the data sources that inform infrastructure and essential services for the IW TWG (TRAC-FLVN, Argonne, TAMU, etc.).
  - Manipulate parameters in SIM to best represent the infrastructure and essential services per TWG representation and integration requirements.
- Output Familiarization and Analysis Development.
  - Become familiar with base set of SIM output files from TWG11.
  - Understand the types of output files that SIM generates and how to produce them.
  - Understand data reduction and manipulation techniques (and automation techniques) to best put the data into a form that suits the needs of the analysis use case.
  - Develop a standard and repeatable analysis approach that fits with IW TWG use case and analysis requirements.

2.6. RECOMMENDATIONS FOR SIM 2.0

All lessons learned from SIM 1.0 development and testing carried over to SIM 2.0. The design of SIM 2.0 followed the overarching goal to simplifying the data development and integration process in order to produce traceable, explainable results. The following list of issues and questions highlighted the challenges and opportunities with developing SIM 2.0:

- There is a need to refactor Nexus code into a SimKit Discrete Event Simulation in order to simplify SIM down to a single model.

- The team must maintain the core capabilities of SIM 1.0 population opinions, infrastructure, communications, key leaders, and networks.
- Is the complexity in the cognitive architecture necessary, or is there a subset of components required to produce acceptable results?
- Is there a better way to model OAB that is more intuitive to a player in the IW TWG and explainable to an analyst?
- What results when executing multiple events that affect an agent per day?
- What results when positive and negative events affect an agent over time?
- How does the SME elicitation data affect agents in the model?

Design Point	Survey	Belief		Issue Stance		OAB					Discount	Scripted Action	Infrastructure	SIM Time	Effect Direction
	# Resp.	Adequate	Inadequate	Adequate	Inadequate	NA	NP	N	PP	PA	λ				
1	100	1	99	1	99	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Positive
2	100	99	1	99	1	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Positive
3	100	50	50	50	50	0.5000				0.5000	0.9900	CFOperatesInArea	N/A	140	Positive
4	100	50	50	50	50	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Positive
5	100	50	50	50	50	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Positive
6	100	100	0	100	0	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Positive
7	100	0	100	0	100	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Positive
8	100	1	99	1	99	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Negative
9	100	99	1	99	1	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Negative
10	100	50	50	50	50	0.5000				0.5000	0.9900	CFOperatesInArea	N/A	140	Negative
11	100	50	50	50	50	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Negative
12	100	50	50	50	50	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Negative
13	100	100	0	100	0	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Negative
14	100	0	100	0	100	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Negative
15	100	1	99	1	99	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Positive
16	100	99	1	99	1	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Positive
17	100	50	50	50	50	0.5000				0.5000	0.5000	CFOperatesInArea	N/A	140	Positive
18	100	50	50	50	50	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Positive
19	100	50	50	50	50	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Positive
20	100	100	0	100	0	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Positive
21	100	0	100	0	100	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Positive
22	100	1	99	1	99	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Negative
23	100	99	1	99	1	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Negative
24	100	50	50	50	50	0.5000				0.5000	0.5000	CFOperatesInArea	N/A	140	Negative
25	100	50	50	50	50	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Negative
26	100	50	50	50	50	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Negative
27	100	100	0	100	0	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Negative
28	100	0	100	0	100	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Negative
29	100	1	99	1	99	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Positive
30	100	99	1	99	1	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Positive
31	100	50	50	50	50	0.5000				0.5000	0.1000	CFOperatesInArea	N/A	140	Positive
32	100	50	50	50	50	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Positive
33	100	50	50	50	50	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Positive
34	100	100	0	100	0	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Positive
35	100	0	100	0	100	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Positive
36	100	1	99	1	99	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Negative
37	100	99	1	99	1	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Negative
38	100	50	50	50	50	0.5000				0.5000	0.1000	CFOperatesInArea	N/A	140	Negative
39	100	50	50	50	50	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Negative
40	100	50	50	50	50	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Negative
41	100	100	0	100	0	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Negative
42	100	0	100	0	100	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Negative
43	100	1	99	1	99	0.9900				0.0100	0.0100	CFOperatesInArea	N/A	140	Positive
44	100	99	1	99	1	0.0100				0.9900	0.0100	CFOperatesInArea	N/A	140	Positive
45	100	50	50	50	50	0.5000				0.5000	0.0100	CFOperatesInArea	N/A	140	Positive
46	100	50	50	50	50	0.9900				0.0100	0.0100	CFOperatesInArea	N/A	140	Positive
47	100	50	50	50	50	0.0100				0.9900	0.0100	CFOperatesInArea	N/A	140	Positive
48	100	100	0	100	0	0.0100				0.9900	0.0100	CFOperatesInArea	N/A	140	Positive
49	100	0	100	0	100	0.9900				0.0100	0.0100	CFOperatesInArea	N/A	140	Positive
50	100	1	99	1	99	0.9900				0.0100	0.0010	CFOperatesInArea	N/A	140	Negative
51	100	99	1	99	1	0.0100				0.9900	0.0010	CFOperatesInArea	N/A	140	Negative
52	100	50	50	50	50	0.5000				0.5000	0.0010	CFOperatesInArea	N/A	140	Negative
53	100	50	50	50	50	0.9900				0.0100	0.0010	CFOperatesInArea	N/A	140	Negative
54	100	50	50	50	50	0.0100				0.9900	0.0010	CFOperatesInArea	N/A	140	Negative
55	100	100	0	100	0	0.0100				0.9900	0.0010	CFOperatesInArea	N/A	140	Negative
56	100	0	100	0	100	0.9900				0.0100	0.0010	CFOperatesInArea	N/A	140	Negative

Table 2.4: Fifth Iteration: Baseline Test Exemplar

3. SIM 2.0

The key design change in SIM 2.0 is the integration of key leader and social networks into the population model. Previously SIM used Nexus Network Learner for key leaders and social networks. Augustine Consulting Incorporated (ACI) contractors implemented Nexus using Java Repast libraries. The reliance on two separate simulation models in SIM was not only inefficient; it required additional coordination, configuration management, and contract dollars to maintain. The transition team's intent for SIM 2.0 was to consolidate the capabilities of both models into a single, Java SimKit-based discrete event simulation. The resulting model reduced complexity in scenario design, decreased SIM execution time, eliminated the need for communication between two separate models, and reduced reliance on contractor support.

3.1. SIM 2.0 BASICS

3.1.1. Population Model

The Social Impact Module version 2.0 is a single model containing a population model and the ability to represent key leader and social networks. As with SIM 1.0, each agent has a set of demographic dimensions that collectively inform the agent's beliefs, values, interests, stances on issues, and behaviors. The narrative paradigm remains the underlying social theory where narrative identities form agent beliefs, values, and interests. SIM 2.0 data requirements remain the same as the Cultural Geography model in SIM 1.0. Once the population is partitioned, scenario developers map beliefs from the available population data using BBNs. The conditional probability tables (CPT) in the BBN are learned from survey data. Analysts use the survey instrument's questions to inform agent's beliefs and interests. SIM 2.0 updates these beliefs over time as part of model execution with minor differences from SIM 1.0.

Appendix E contains a description of improvements made in SIM 2.0 to address specified requirements. Appendix E focuses on the key leader and social network additions to SIM since the population model in SIM 2.0 closely resembles the SIM 1.0 population model, described in Appendix C. The following list summarizes the major SIM 2.0 population model improvements:

- Finalized the implementation of improved communications introduced during the testing of SIM 1.0.
- Verified minimal impacts of Recognition Primed Decision Making (RPD) and Trust modules on outputs thus allowing the team to recommend turning these modules off for the IW TWG use case.
- Fixed minor issues identified during testing and evaluation of the population model

in SIM 2.0.

- Identified best practices for scenario development and variable assignment in the scenario file for TRAC-WSMR IW TWG team.

3.1.2. Key Leader Engagements

The most significant difference between SIM 1.0 and SIM 2.0 is the addition of key leader and social network components. These refactored Nexus components enable actors in the IW TWG to conduct simulated key leader engagements. KLE can result in wargame participants gaining useful information such as knowledge about key leaders, threat networks, or general knowledge contained in the TWG database. SIM 2.0 also develops and updates the static and dynamic networks within the simulation. These networks model the social, professional, personal, criminal, and threat networks that exist within a population.

Networks in SIM 2.0 are static or dynamic. The difference between static and dynamic networks merits some discussion. By definition, a static network is one that does not change in the game. An example is a terrorist cell modeled by scenario developers using actual threat network data. Most likely this data will be classified and represent an actual network in an area of operation. The IW TWG team identifies potential threat networks and leverages the appropriate data to model the network using the scenario file prior to game time. The file instantiates the network in the model on initialization and it remains in place throughout the game. In contrast, a dynamic network changes during the game. One use for the dynamic network is to model human relationships, such as marriage or divorce. Those create family networks that can change or dissolve. Roles define the networks in the model and determine with whom agents communicate, delineate what agents know, and establish the range of possible outcomes that might result from engagements.

SIM 2.0 contains the ability to remove key leaders from their static networks. Removal can occur by capturing or eliminating a key leader. Model instructions define the results of these player actions. The model also has the capacity to conduct a Shura, town hall meetings, or any other key leader gathering. The name is not important and scenario designers establish what these meetings are called. Properly defining areas of operation for players, tribal boundaries, and the base hexagons in the game determine what key leaders will attend requested meetings. All leaders in a specified area will attend the meeting; however, the scenario designer can specify in the scenario file the probability that leaders will choose not to attend. See Appendix H for specific variable settings for these results.

3.2. SIM 2.0 MODEL INTEGRATION

3.2.1. Interaction with PAVE

The SIM 2.0 scenario file contains a PAVE interface scenario worksheet named "PaveInterface". In the worksheet a scenario developer specifies a PAVE database by name and location (path). This is unchanged from SIM 1.0 and the team still recommends conducting a warm-up period due to continued use of BBNs.

After the warm-up period, the wargame begins and players can input their actions into PAVE. Unlike SIM 1.0, there is no need to run separate population and key leader models. Once wargame players input their objectives into PAVE, actions and events occur in SIM chronologically based on scheduling. SIM writes the results of each turn back to the PAVE database upon completion of a model run. This requires significantly less coordination and greatly reduces the time required to produce population, key leader, and social network results from the model.

3.2.2. Integration Testing

Testing of SIM 2.0 continues as the authors produce this report. In concert with incremental developments to PAVE, SIM-PAVE integration testing occurs continuously to ensure the changes to both models achieve IW TWG objectives without any loss of information. TRAC-WSMR will need to continue testing SIM 2.0 integration after transition due to scheduled development of PAVE during FY13. This requires aggressive configuration management (CM) controls that have been discussed often at the Models Integrated Product Team (IPT) teleconferences.

Integration testing occurred using the latest copies of the PAVE database. Integration testing also took place in the DISA RACE environment. These tests mirrored the SIM 1.0 testing methodology where controlled inputs were used in order to verify specific outputs from the model. Small-scale SIM 2.0 tests verified that model instructions in PAVE executed properly in SIM 2.0. The first major integration testing event took place during SIM 2.0 training at TRAC-WSMR during the week of 17-21 September 2012. The team used a small-scale scenario file prepared by TRAC-MTRY for training. This testing verified that SIM produced all required PAVE data for the IW TWG. The training team also created a Shura request testing scenario on site that produced desired results and a simple fix to the manner in which SIM wrote the results to the database.

3.3. SIM 2.0 DESIGN

3.3.1. Scenario Files and Data

The additional capability of SIM 2.0 necessitated a change to the scenario file for SIM. The primary change included the addition of thirteen (13) worksheets needed for key leader and network instantiation in SIM. A small Extensible Markup Language (XML) file defines how the key leader and social networks components handle various model instructions such as key leader removal or Shura (key leader meeting) requests. Appendix H describes how to create key leaders and implement static and dynamic networks in SIM 2.0 and describes of the XML file required for SIM 2.0.

3.3.1.1. Inputs

The Microsoft Excel scenario file used by SIM 1.0 established the foundation for SIM 2.0 scenario work. The population model input data required by SIM 2.0 remains intact. Discussion of the complete scenario file is beyond the scope of this report; however, Appendix H contains the SIM User Guide and explains each variable required to build a SIM 2.0 scenario. This list highlights some of the key additions to a SIM 2.0 scenario file.

- A worksheet defining social network behaviors. For example, SIM entities are able to marry, divorce, and tell stories. These behaviors create “noise” for the social model that begin and end specified behaviors.
- Roles in both the static and dynamic networks are outlined in one worksheet dedicated to establishing all direct and derived relationships.
- The new “Role Behaviors” tab enables the scenario designer to specify what roles are able to perform named behaviors.
- Role qualifications empower developers to list the required, desired or disaggregated relationships between a role and its dimension-value pair.
- The “Key Leader Network” tab provides a single location for listing the name of each key leader network, leader, roles and subordinates in the network.
- Affinity levels define distances between affinity states. Generically, a level 1 affinity represents a key leader who does not like a player at all and will go out of his way to lie and deceive a player. A level 5 affinity represents a key leader that will fully cooperate with a wargame player’s requests.
- A “Light Entity Prototype” worksheet adds the ability to create the agents that will participate in the dynamic network. Scenario designers declare the quantity, names, and locations of these entities.

3.3.1.2. Outputs

After a model run SIM 2.0 outputs a series of comma separated value (.csv) files using the same method as SIM 1.0 for the population model. Loggers in the model determine the data written to the output files. SIM writes the logged data to files specified in the scenario file. The number of output files depends on the number of issues, actors, key leaders, networks, and knowledge in the wargame. For the population model each issue has an associated file for tracking population issue stances. Similarly, actors each have an associated output file for recording the OAB changes pertaining to that actor. When used in a wargame SIM 2.0 required outputs are population, key leader, and network data to the PAVE database. Software developers can adjust what data goes to the database and future IW TWG requirements may require new data logging. Currently the IW TWG use case only writes key leader and social network data to PAVE since there has been no requirement to write separate files for analysis.

3.4. SIM 2.0 TESTING

3.4.1. Objective

SIM 2.0 testing verified that the model works correctly and that all SIM 1.0 functionality remained in the latest version. Integration testing continued to focus on ensuring that SIM 2.0 was interoperable with PAVE and thus produced outputs necessary for conducting a future IW TWG. It is worth reemphasizing that configuration management of the models in the IW TWG is critical after the transition of SIM from TRAC-MTRY to TRAC-WSMR. The team conducted testing based on a requirement to replicate the behaviors of past wargames; it is quite possible that an unidentified need could arise during the development of the next IW TWG objectives that SIM 2.0 is not prepared to accomplish.

3.4.2. Overview

The testing of SIM 2.0 began in May 2012. The team began by executing all the fifth iteration tests from SIM 1.0. These tests ensured that SIM 2.0 produced similar outputs. A Python script developed at TRAC-MTRY compared the output data and highlighted any differences between SIM 1.0 and SIM 2.0 output files when using the same scenario file. No differences in the resulting data demonstrated that the population model maintained all capabilities from the stabilized version of SIM 1.0. The remainder of this section covers the methodology for testing SIM 2.0. See Appendix F and Appendix G for more analysis of the resulting data. All test data for SIM 2.0 resides on the DVDs delivered to TRAC-WSMR on 21 September 2012.

3.4.3. Iterations

3.4.3.1. First iteration: Interesting Results from SIM 1.0

After executing the SIM 1.0 test cases in SIM 2.0 the team focused on a set of nine test conditions that produced results displaying the greatest population opinion changes in the model. SIM 1.0 identified that the difference between 100 and 10,000 survey respondents was statistically insignificant; however, decreasing by an order of magnitude from 100 did produce measurably inferior results. The team demonstrated this result by testing a stereotype that had only 10 survey respondents. Initially the team decided that 100 respondents per case file was an acceptable target for stereotype development and testing.

Additionally, any discount factor (λ) greater than 0.1 caused the model to run "too hot" producing results too quickly. Due to high discount factors, agents "forgot" things that happened in their recent past and based their issue stances and OAB on the most recent set of circumstances. Therefore, the team only used discount factors of 0.1 and 0.01 for testing scenarios with a length of 140 simulation units. Finally, the positive and negative effects of the scripted actions varied the results in different directions, based on the mapping in the scenario file. Table 3.1 outlines the nine (9) primary test conditions for SIM 2.0's first iteration of tests:

	Design	Survey	Belief		Issue Stance		OAB		Discount	Scripted Action	SIM Time	Effect	Reason for Inclusion
#	Point	# Resp.	Adequate	Inadequate	Adequate	Inadequate	NA	PA	λ			Direction	
1	29	100	1	99	1	99	0.9900	0.0100	0.1000	CFOperatesInArea	140	Positive	Recommended min of 100 repondents; λ=0.1; spread of initial beliefs on **negative** extremes; Positive event.
2	32	100	50	50	50	50	0.9900	0.0100	0.1000	CFOperatesInArea	140	Positive	
3	35	100	0	100	0	100	0.9900	0.0100	0.1000	CFOperatesInArea	140	Positive	
4	43	100	1	99	1	99	0.9900	0.0100	0.0100	CFOperatesInArea	140	Positive	Recommended min of 100 repondents; λ=0.01; spread of initial beliefs on extremes; Positive event.
5	46	100	50	50	50	50	0.9900	0.0100	0.0100	CFOperatesInArea	140	Positive	
6	49	100	0	100	0	100	0.9900	0.0100	0.0100	CFOperatesInArea	140	Positive	
7	51	100	99	1	99	1	0.0100	0.9900	0.0100	CFOperatesInArea	140	Negative	Recommended min of 100 repondents; λ=0.01; spread of initial beliefs on **positive** extremes; Negative event.
8	54	100	50	50	50	50	0.0100	0.9900	0.0100	CFOperatesInArea	140	Negative	
9	55	100	100	0	100	0	0.0100	0.9900	0.0100	CFOperatesInArea	140	Negative	

Table 3.1: Nine SIM 1.0 Results for Additional SIM 2.0 Testing

Table 3.1 outlines the three primary tests. The first three tests demonstrate that positive events have a significant effect on population agents with an extremely negative opinion. The middle three highlight the effect of changing the discount factor on how quickly the results occur. The final three tests confirm that negative events causes a decline in issue stance for population agents. For the last three test cases, the belief, issue stance, and OAB changed from very positive initial stances. This allowed the team to analyze the decrease in opinion over a longer period of time. If scenario developers used the same negative case files employed in the first six tests, the results would have shown very little movement because the issue stance already contained almost no evidence of adequacy. The team built each of these nine variable combinations into separate scenario files establishing a new baseline set of experimental design points.

After setting a SIM 2.0 baseline, the team needed to test the recommended minimum threshold of 100 survey respondents for developing the population stereotypes. SIM 1.0 testing focused on looking at the differences between 100, 1000, and 10,000 survey respondents for a given stereotype. This was useful to show that there is not a significant benefit beyond 100 respondents; however, the team wanted to refine the number for the recommended minimum number of survey respondents. An initial test done during SIM 1.0 training revealed that only using 10 survey respondents produced a statistically significant result. The development team created another DOE to vary the number of survey respondents. These new DP aimed to determine where there was a significant decline in population opinion results due to inadequate evidence from survey results. The team chose to test 10, 50, and 75 respondents using the conditions listed in Table 3.1. The results showed that 100 respondents are significantly different from 10 respondents at $\alpha = 0.01$. Using 75 survey respondents is statistically significant when compared to 10 respondents at $\alpha = 0.05$. The design points with 50 respondents were not significantly different from 10 survey respondents. It is important to understand that the larger the BBN, the more evidence the BBN requires to avoid "coin flip" results. Evidence for every combination of beliefs and interests is ideal, but the data may not contain all combinations. This is where the analyst must make decisions about how to best represent the population. Using 75 respondents may be appropriate in some situations; however, from this point on, the research team recommends using a minimum of 100 respondents for any population stereotype as a general rule.

3.4.3.2. Second iteration: Adding Complexity to a Scenario

After the team established a new baseline it was time to add complexity to the scenario files to test more complex situations. The factors added to the model were:

- Multiple actions per day.
- Alternating occurrences of positive and negative actions.
- Reducing the effects of actions on population agents. Previously the team controlled effects by mapping them to 100% positive perceptions. This new design changed that mapping to a 75% positive effect providing a slower change in population agent opinions.
- Minimizing the effects of actions to 50% positive effects thus essentially making agent reactions a coin flip.

3.4.3.3. Third iteration: Applying Techniques to a Specific Region

During SIM Transition In Progress Review (IPR) #2 in June 2012, the team asked about potential study locations for the next IW TWG. One possibility was a scenario in Africa. At the time, TRAC-MTRY was working on another project aimed at analyzing survey data from the Sahel region of Africa. The transition team suggested evaluating this Africa survey data using SIM. TRAC-WSMR agreed that using the data might prove useful. After the IPR the team spent the next month building the necessary case files and using those case files with scenario files already developed for testing.

The scenario files used for testing the Africa data were the most recent (second) iteration of test files. The team used the scenario files for multiple actions, alternating actions, 75% positive effects, and 50% positive effects with the Africa population case files. The different case files were the sole variable changed in this iteration. TRAC-MTRY MAJ Deveans divided the population agents among three demographic dimensions containing two factors each:

- Gender: Male or Female.
- Religion: Christian or Muslim.
- Age: Over-30 or Under-30.

These demographic dimensions created eight (8) population agent prototypes and ensured a significant number of respondents for each issue. The 8 prototypes are below with the number of respondents in parenthesis after the stereotype abbreviation.

- F_C_O (419): Female Christians Over-30.
- F_C_U (602): Female Christians Under-30.
- F_M_O (309): Female Muslims Over-30.
- F_M_U (498): Female Muslims Under-30.
- M_C_O (481): Male Christians Over-30.
- M_C_U (578): Male Christians Under-30.
- M_M_O (388): Male Muslims Over-30.
- M_M_U (462): Male Muslims Under-30.

The team intended to use of these stereotypes as a proof-of-principle demonstration that other stereotypes from another region would work in SIM 2.0. As it turns out, many of these stereotypes demonstrated remarkably similar opinions over time. The team did not conduct SME elicitation to determine the effects that potential scenario events might have on the population. Instead the team used the same effects as those used for the second iteration of SIM 2.0 testing. Shortly after completing these tests interest in this data waned and the team refocused on data from the Afghanistan surveys used for the IW TWG 2011.

While the team did not conduct a lot of tests on this alternate set of data, the results confirmed what the team expected to see–that the effects table of the scenario file has a substantial impact on the outputs from the model. Even though the team used population agents from a different part of the world, the model produced similar results to previous tests. This result proved that analysts can trace results back to inputs in the scenario file and it demonstrates that those results are explainable. See Appendix F for an overview of these test results.

3.4.3.4. Fourth iteration: Evaluating the Cognitive Architecture.

In order to simplify SIM 2.0, a detailed look at the population model's cognitive architecture was necessary. Specifically, the team suspected that the RPD and trust modules did not provide a significant effect on outputs from the model. Using both RPD and Exploratory Learning (EL) requires extra loggers for analysts to trace results from the model. Agent decisions using RPD vary slightly from those employing an EL. Furthermore, the use of these modules might create additional complexity without any benefit.

To answer these questions, the team recruited MAJ Chin Chuan "Chase" Ong from the Singapore Armed Forces. Major Ong approached TRAC-MTRY seeking a thesis topic for his Masters of Science (MS) in Modeling, Virtual Environments, and Simulation (MOVES). He sought a topic involving agent behaviors and the SIM Transition team guided his research. Major Ong's results greatly assisted the team in deciding what components to recommend using and informed modifications to the final version SIM 2.0.

After testing over 30,000 replications in the cognitive architecture, the team is able to say with confidence that the use of both RPD and EL in the cognitive architecture provides no statistically significant advantage. Results from both modules end up with nearly the same end state or agent decision. Similarly, the only difference between using and not using the trust module is additional variance. It provides no additional benefit to model outputs, but does create additional overhead in the model. Therefore, the team recommends using EL only with the trust module turned off.

In addition to the evidence about RPD and EL, Major Ong's testing also uncovered a rare situation involving scenario event effects and population opinions. When LTC Caldwell and Major Ong examined the data there was a design point where a population agent had a 99% adequate view of civil security. While infrastructure was damaged, his opinion decreased. After a set amount of time, there was a repair event for the damaged infrastructure. Even after this repair the agent's opinion continued to decrease. When that test concluded, the agent opinion had decrease to a belief that civil security was only 40% adequate. After examining the code and reviewing the underlying equations for this calculation, the team verified that the model behaves properly. However, future scenario designers must be aware that positive effects must be greater than the initial issue stance of agent stereotypes when the simulation begins. For instance, the agent described above with an initial civil security issue stance can only increase this issue stance if the effect is greater than 99%. Of course this is an extereme case, but it equates to the belief that anything less than perfection will disappoint the agent. Given the discount factor (λ) of 0.01, the agent will remember failure for 100 time units, and the TWG will most likely be over before the agent's opinion can ever be turned around.

Appendix G contains Major Ong's complete thesis. The development team delivered his experimental design spreadsheet, input scenario files, output (.csv) files, and analysis to TRAC-WSMR during the SIM 2.0 Transition Training in September 2012.

3.4.4. Significant Findings

Appendix F and Appendix G contain test results and analysis on SIM 2.0, and the team delivered all testing scenario files and output files to TRAC-WSMR with the software. A summary of the most significant findings follows.

- Event effects are most significant further away from a 50-50% effect.
- Multiple events occurring in a single SIM day have a significant effect over time, especially with a lower discount factor (λ).
- RPD and trust modules provide minimal effect on model outputs.
- Scenario designers must ensure that effects intended to be positive have an effect value greater than the initial issue (adequate) opinion of the population agent.

3.5. SIM 2.0 TRAINING

3.5.1. Overview

SIM 2.0 training occurred during the week of 17-21 September 2012. The team began the training sessions by reviewing the population scenario file worksheets from SIM 1.0. These worksheets had only minor changes from SIM 1.0. For Ms. Clark this was a review of her training with TRAC-MTRY. New trainees learned how to properly prepare the population worksheets. The primary focus of the remainder of SIM 2.0 training covered the 13 new worksheets that detail the key leader and social network implementation. Conveniently, the PAVE developers attended the training and the team placed special emphasis on locations in the scenario file where SIM data must align with data in PAVE. Additionally, the training group observed the execution of 3-4 test cases involving key leader engagements, dynamic social network activity, critical knowledge, and a Shura request.

3.5.2. Objectives

The training accomplished the following objectives:

- Provide an overview of SIM 2.0, event graphs, and discrete event simulation modeling.

- Review 61 population model worksheets.
- Discuss the structure of the key leader and social network model.
- Conduct overview of 13 additional worksheets for key leader and social network model implementation.
- Conduct practical exercises focused on the proper use of worksheets in the scenario file.
- Connect SIM 2.0 to PAVE and execute model runs using a complete, simple scenario file with PAVE.
- Review documentation for SIM 2.0.

After accomplishing the training objectives, the team discussed an alternate version of SIM 2.0 developed by TRAC-MTRY. This alternate version, SIM 2.0a, was not requested by TRAC-WSMR; however, the transition team wanted to explore the possibility of simplifying OAB. SIM 2.0a reduces OAB to a set of state variables, or counters, that keep track of positive and negative events that affect population agents. If something positive happens to an agent, 1 unit gets added to their OAB. Oppositely, if something negative happens to an agent, 1 unit gets subtracted from the agent's OAB. The IW TWG team can decide how to bin and display these OAB, if they decide to use them. Using state variables to represent OAB simplifies the use of OAB and makes the model more explainable than the use of five (5) state probabilities for OAB as used in previous wargames. The development team does not believe that the 5-state OAB are intuitive to a human player and have the potential to confuse wargame participants. Furthermore, using a counter system is akin to everyday measurements, such as a fuel gage or a bank account where withdrawals and deposits are made that reflect positive and negative actions.

TRAC-MTRY delivered SIM 2.0a to TRAC-WSMR on 18 September 2012. Currently SIM 2.0a is not compatible with PAVE and would require minimal changes to PAVE. Developed as a proof-of-principle simplification, SIM 2.0a is stable and ready for use should TRAC-WSMR decide that it is a more suitable solution for population modeling.

3.6. RECOMMENDATIONS FOR SIM 3.0

After completing SIM 2.0, the team reviewed the overall objectives of SIM Transition. Recall that TRAC-WSMR desired a simple, explainable model with traceable results. The testing conducted on SIM 2.0 clearly demonstrated that SIM 2.0 results were explainable and traceable back to the data used to develop the scenario file. Early in the transition process the team identified a simpler and more explainable approach to modeling a population and this process formed the basis for SIM 3.0. The intent behind SIM 3.0 was to think “outside the box” in order to produce a model true to the theoretical underpinnings of SIM, but one with more explainable model results even with increased complexity.

4. SIM 3.0

During SIM 1.0 stabilization efforts, the transition team also began exploring alternatives to Bayesian Belief Network modeling techniques. Immediately, the team identified Markov Chains as another method of modeling discrete state probabilities. Although a Markovian approach seemed appropriate, the team realized that it did not simplify data requirements or minimize SME elicitation. As a result the team continued investigating and evaluating other potential methods.

Fortuitously, TRAC-MTRY worked on a project concurrent to SIM Transition sponsored by the Center for Army Analysis (CAA). This project's was Africa Knowledge, Data Source, and Analytic Effort (KDAE) Exploration. Part of the Africa KDAE research developed a methodology and built a proof-of-principle scenario in a specific region or country in the United States Army (USA) Africa Command (AFRICOM) area of responsibility (AOR) for use in future IW TWGs using Factor Analysis and Generalized Linear Models (GLM). Appendix J contains an excerpt from the KDAE technical report describing the data development process in more detail. It provides a practical example for developing the underlying models supporting a SIM scenario file.

4.1. SIM 3.0 BASICS

The Social Impact Module version 3.0's data development methodology for population modeling changes significantly in this version compared to previous versions of SIM. The scenario file contains eight (8) fewer worksheets, removing many of the belief and issue related input because BBN are no longer used for agent issue stances and OAB. SIM 3.0 still contains the ability to represent key leaders and social networks and there are no changes to the procedures described for SIM 2.0 regarding key leaders and networks.

4.1.1. Population Model

As with previous versions of SIM, each agent has a set of demographic dimensions that collectively shape the agent's beliefs, values, interests, stances on issues, and behaviors. TRAC-MTRY highly recommends the use of SMEs to assist with developing the cultural narratives that define each population stereotype. Ensuring that each stereotypes has a large number of survey respondents is also a recommended practice based on the testing done with previous versions of SIM. The development team continues to recommend a minimum of 100 survey respondents per stereotype.

The significant change in SIM 3.0 is the data-driven approach to modeling the population. The methodology results in a series of look-up tables that determine an agents issue stance and OAB based on a specific event. When the event occurs, SIM looks up the agent's new issue stance and OAB from the tables and logs the change. Loggers can track the delta (Δ)

between these look-up values over time to show the change in population agent opinions. SMEs assist scenario developers by providing expert opinions about the directional effect that each PAVE event will have on a given population agent. These effects are a key component in generating the models that define the lookup tables. See Appendix J for a complete overview of the data development process to include R code needed to prepare survey data for table development.

4.1.2. Key Leader Engagements

The same 13 worksheets define key leaders, social networks, roles, behaviors, and the variables required to execute these components in SIM. There are no changes to the SIM 2.0 software or scenario file for KLE components in SIM 3.0.

4.2. SIM 3.0 MODEL INTEGRATION

As of this report SIM 3.0 has not been tested with PAVE. Due to the "new" techniques used in SIM 3.0 there would need to be minor modifications made to the PAVE database. These include changing the way OAB are tracked and displayed much like the OAB alternatives explored in SIM 2.0a. During the training and integration event at TRAC-WSMR in September 2012, PAVE and SIM developers discussed the changes that would be required to the database. The changes would be relatively minor requirng only new fields for the new issue stance and OAB results.

4.3. SIM 3.0 DESIGN

The design of SIM 3.0 varies little from SIM 2.0. Instead of using the cognitive architecture to adjudicate what actions the SIM agents take new classes implement a method to use the lookup tables created for agent issue stances and OAB. SIM 3.0 does not eliminate the need for the cognitive architecture. The cognitive architecture still determines how agents seek their infrastructure needs. It also handles the decision by agents to communicate using the Action Selection Module (See Appendix C). However, once the communication percepts reach another agent the look-up tables determine the effects on those agents receiving the communications.

4.4. SIM 3.0 TESTING

SIM 3.0 testing consisted of JUnit Testing in Java to verify that the agent issue stance and OAB updates utilized the lookup tables produced during data development. The team developed a few proof-of-principle scenario files for testing and training. The team delivered these scenario files to the TRAC-WSMR IW TWG team for future use, testing, and exploration.

4.5. SIM 3.0 TRAINING

4.5.1. Overview

SIM 3.0 training occurred the same week as SIM 2.0 training, 17-21 September 2012. The team focused initially on changes to the scenario file, highlighting the worksheets no longer in the scenario file and those worksheets whose columns were modified. Once the overview of the scenario file was complete the team shifted to the primary effort of SIM 3.0 training–the data development process. MAJ Deveans taught trainees how to employ the methodology developed as part of the KDAE project. He explained how the resulting models contain the survey questions that inform issue stances (factors). Next participants learned that scenario developers should gather SME input that determines how events affect population agents positively, negatively, or with no effect. Combined this produces a set of look-up tables by stereotype containing the effects of each possible scenario event.

4.5.2. Objectives

The training accomplished the following objectives:

- Review new scenario file.
- Overview of the data development process.
- Conduct practical exercises focused on the proper data development process, building scenario files, and executing SIM with a complete, simple scenario.
- Review documentation for SIM 3.0.

4.6. RECOMMENDATIONS FOR FUTURE DEVELOPMENT

SIM 3.0 is an experimental model based on a new methodology for survey data analysis to inform scenario file development. It is possible that the IW TWG team will want additional development of SIM 3.0. The data development process employed in SIM 3.0 could portentially inform data development for SIM 2.0. This was accomplished by the TRAC-MTRY SIM Transition team during testing of SIM 2.0. Recall that one of the SIM 2.0 testing iterations used data from the KDAE project. The population case files and issue stance data for the BBN came from the same data set used in the KDAE project. The team briefly explored this hybrid of data development techniques used by SIM 2.0 and SIM 3.0, but it merits future refinement and development of best practices before employing in the IW TWG.

During SIM 3.0 training, the PAVE representatives noticed that many of the data development activities for SIM scenario files could benefit from the use of a database. A database would reduce the repetitive mapping of multiple beliefs, issues, agents, and events, thus reducing the time required for scenario development. It also has the potential to eliminate a significant amount of worksheets required in the scenario file. Finally, all SIM data could potentially reside within PAVE, providing the analyst access to additional data.

5. CONCLUSION AND RECOMMENDATIONS

5.1. FINDINGS

5.1.1. Iterative Approach to the Design of SIM

Using an iterative approach to the systems design process, the SIM Transition team identified stakeholder requirements, proposed potential solutions, gained acceptance for recommended solutions, implemented those solutions and elicited stakeholder feedback. The methodology employed resulted in three major versions of SIM over Fiscal Year (FY) 2012 designed specifically for the IW TWG. First, the stabilized the version of SIM used in the IW TWG 2011 became SIM 1.0. SIM 1.0 contained two models: Cultural Geography and Nexus Network Learner implemented in two different Java-based libraries. SIM 1.0 represented a good start, but there was significant work needed to improve the models based on stakeholder feedback from the IW TWG team. WSMR requested that future versions of SIM not only be stable, but those versions also produce results that are explainable and traceable.

These requests prompted the TRAC-MTRY development team to combine the two models into a single Java SimKit-based Discrete Event Simulation, SIM 2.0. SIM 2.0 reduced challenges created by having two separate models. It also made the social model more explainable by reducing the amount of integration required to execute model runs in the IW TWG. With SIM 2.0 PAVE only has to communicate with one model instead of two and multiple events occur in a single model run to include population opinion updates, key leader engagements, and social network activity. During the development of SIM 2.0, the TRAC-MTRY team investigated alternative ways to represent OAB. This was not a specific requirement for SIM Transition; however, a review of the data development process revealed that issue stances, not OAB, are the most significant result from SIM while OAB are the primary result communicated to TWG players about how they are performing in the wargame.

Finally, SIM 3.0 leveraged other project work at TRAC-MTRY to develop a new data-driven methodology for SIM scenario design. The resulting models produced look-up tables for use in SIM. These look-up tables further simplify SIM and the results are the most traceable and easily explainable yet. Tables derrived from the SIM 3.0 data development methods provide issue stance and OAB values for each event.

5.1.2. Testing

There are numerous findings from testing SIM. This section will focus on those results that significantly impact model results. SIM 1.0 and SIM 2.0 testing revealed interesting insights on parameters in the model. Most significant among them is the discount factor (λ) used to determine the rate at which previous events are discounted by agents in the model. Figure 5.1 shows that a discount factor of 0.01 creates statistically significant results over time (DP_43) when compared to a discount factor of 0.99 (DP_1), 0.5 (DP_15), and 0.1 (DP_29). A lower discount factor ensures that agents retain evidence of positive and negative events. Wargame players who continue to conduct positive operations will experience a positive result over time because the evidence will mount and will not be "forgotten" by agents in SIM. With a high discount factor, agents marginalize events of the past and are quickly influenced by the event happening right now. This may sound like a good way to get a quick output from the model, but negative events affect agents just as quickly. The result will most likely be a combination of positive or negative events that average somewhere between the extremes.

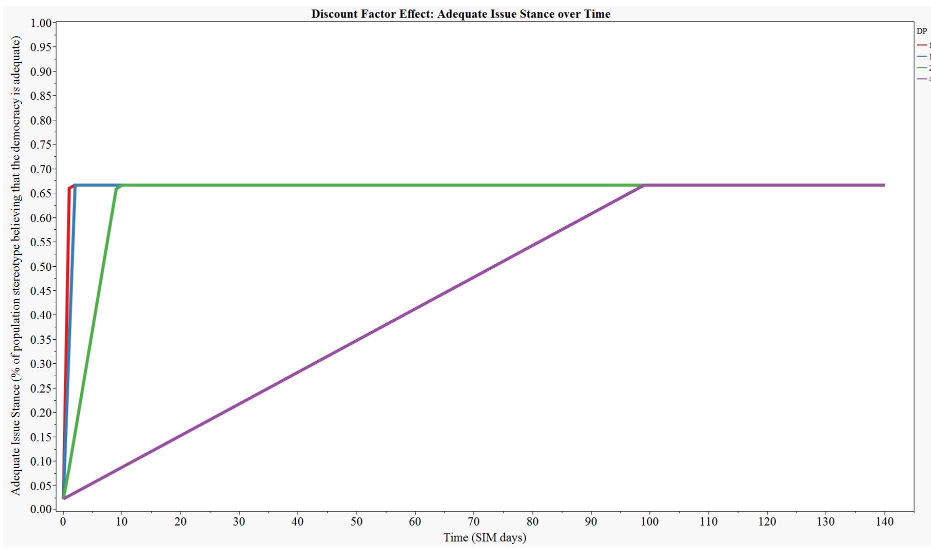

Figure 5.1: Effects of the Discount Factor on Population Opinions

Next, SIM users develop population stereotypes based on demographic dimensions reported in survey data for a region of interest. This underlying survey data has a profound effect on the model. In past TWG scenarios there were numerous stereotypes that lacked evidence in the BBN for the beliefs that inform population issue stances. Often this lack of evidence is a direct result of too many demographic dimensions that result in stereotypes formed by only a few survey respondents. This has the effect of relegating issue stance and OAB changes to the flip of a coin, and it occurs because the BBN lacks data to represent a specific combination of beliefs. Without data, BBNs substitute equal probability for the blanks in the parent (belief) nodes. This happens most frequently when there are not enough respondents to provide a rich mix of responses. Insufficient data results in poor outputs from the BBN that appear to stabilize in equally likely occurances of beliefs. For issue stances, for example, population agents gravitate towards 50% adequate and 50% inadequate. For OAB agents stabilize near a 20% chance of being in one of five states: Positive Passive, Positive Active, Neutral, Negative Passive, or Negative Active. This is not a useful result. Figure 5.2 shows how OAB initialized to extremes stabilize near a 20% chance of being in any OAB state due to a lack of survey repondent evidence in the case files.

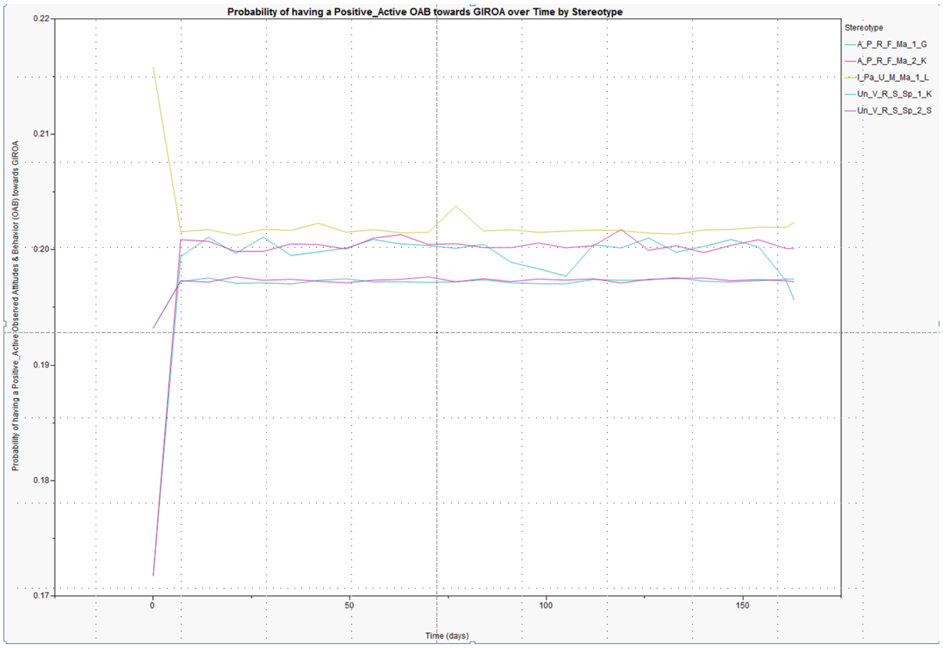

Figure 5.2: Survey Respondent Impacts: OAB for 5 Agents in TWG11 Data

The SME elicitation data is as important as the underlying survey data. SMEs complete surveys that inform how events in the model effect agents in the model. The closer these estimated effects come to equal outcome probabilities, the more likely SIM will produce uninteresting results. For example a SME may provide input on how a population stereotype will perceive an event. If the SMEs say that it will affect 55% of the stereotype positively and 45% of the stereotype negatively, this centers around 50%. This effect may be completely accurate; however, it relegates SIM to essentially flipping a coin to adjudicate the issue stance. It creates an expected values akin to chance and results in issue stances that stabilize around 50-50 mixes. For a HITL wargame this does not help players make decisions and it does not provide a player good feedback on their actions. Figure 5.3 shows five design points tested using the KDAE dataset. These design points were:

- DP_225: Baseline effects of an event that the population perceives as 100% positive.
- DP_243: Multiple, repetitive events per day.
- DP_261: Both positive and negative effects occurring each day.
- DP_279: Effects of an event that the population perceives at only 75% positive.
- DP_297: Effects viewed as 50% positive and 50% negative (50-50).

Notice that effects near 100% positive climb sharply. Effects that are 75% positive climb steady, and effects near 50% move toward the 50% adequate line. There are additional factors with the 50-50 data. See Appendix D for additional information about these design points.

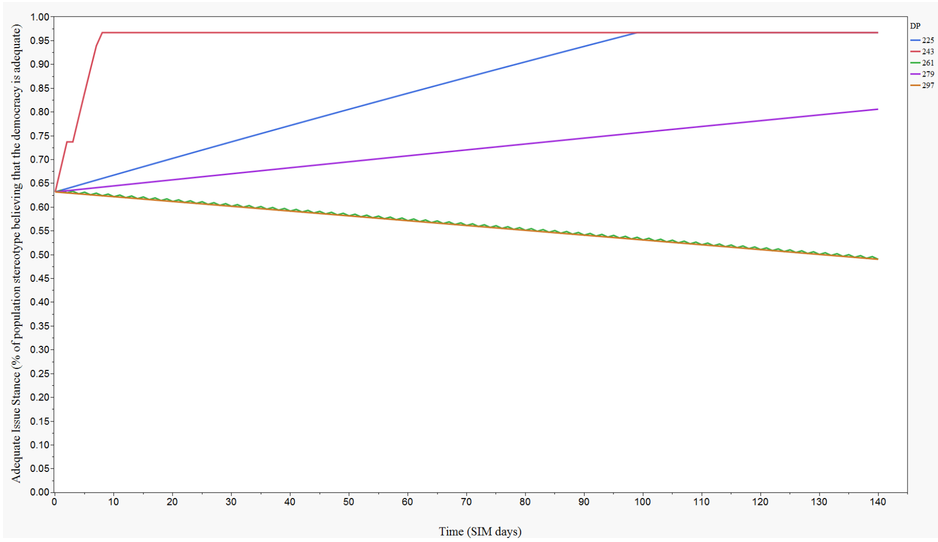

Figure 5.3: Varying Effects and Numbers of Actions

5.2. RECOMMENDATIONS

There are numerous recommendations discovered while developing, documenting, and testing SIM. The four primary recommendations for the use of SIM discovered during SIM Transition are:

- Use a discount factor (λ) of 0.01. The discount factor has a significant effect on how long agents remember good and bad events. A lower discount factor ensures that they have a longer “memory”, and will result in better and more rational agent behavior over time.
- Population stereotypes should have around 100 survey respondents per agent stereotype prototype. This results in better underlying data for the Bayesian Belief Networks, and is likely to provide more evidence for all combinations in the conditional probability table.
- Avoid using effects data that centers around 50% for any event. This relegates the effects of scenario events to a coin flip resulting in poor output data from the model.
- Use fewer events or bin similar events to minimize effects in the model. The use of hundreds of events that each carry an effect dilutes the impact of each event and adds unnecessary complexity to the model.

In addition to these four recommendations, the following list outlines other recommendations and best practices for the use of SIM:

- SIM is designed to run less than 2-years of simulated time. It is very capable of running 100-years; however, the development team strongly discourages this type of use. The demographic dimensions modeled in SIM are static, so agents do not get older, change their political views or otherwise "grow" out of a given stereotype.

- SIM works best at the tactical level (Brigade and below). The inputs can be developed for national and regional level modeling with SIM, but some research is necessary since the development team has no experience with SIM at this level.

- Scenario designers should pay close attention to the effects of events on population stereotypes received from SME. One of the primary recommendations was avoiding 50-50% data, but there is another significant issue to guard against. If the effect is less than the initial value (%) issue stance, the issue stance can only decrease, even if it is viewed positively. This is rare, but happened during testing when the team used extreme issue stances of 99% and 100% adequate. Agents will always be "disappointed" because the effect of a positive action is not as great as their instantiation in the model. The converse is true about negative agents and negative results. If the effect is greater than the initial issue stance, the issue stance will only increase even if it has a very low effect.

- SIM is good at modeling issue stances. It can model OAB, but often survey questions do not ask about attitudes (positive passive, negative active, etc.) and instead ask questions about a person's opinion on the issues. If OABs continue to be part of the IW TWG, consider finding data sources that ask questions specific similar to the way OABs will be modeled or creating surveys to develop this data. Other alternatives include using the counter system described in SIM 2.0 and having SMEs determine if an event will have a positive, negative or neutral effect.

- Use of Factor Analysis techniques explored as part of SIM 3.0 development to determine the issues that matter to a modeled population is highly encouraged. Instead of determining a priori what the issues are and forcing population opinion into those bins, use Factor Analysis to allow the data to tell you what is important to the people of a region. These techniques can provide the data needed for SIM 2.0. The design team proved this process works when testing the KDAE data in SIM 2.0.

- The best use of SIM might be to combine the best of different versions. The development team did not have the time or resources to build and test a hybrid configuration; however, SimKit modules are interchangeable. Minor modifications to the SIM code will allow the WSMR team to experiment with these possibilities.

- Conducting a calibration exercise before the the next IW TWG is absolutely essential to getting the model results desired by the TWG team. SIM is extremely flexible and by doing slight modifications to the scenario file, most results can be achieved.

APPENDIX A. SPECIFIED REQUIREMENTS

A.1. OVERVIEW

This Appendix is from the Requirements Document provided to TRAC-MTRY by TRAC-WSMR. WSMR and MTRY iterated on this document between January and May 2012. The authors used additional formatting to convert the landscape Microsoft Word document to the portrait format used in this report.

Purpose:
Document requirements for the Irregular Warfare Tactical War Game (IW TWG) Social Impact Module (SIM) model design/development/test/transfer from TRAC-MTRY to TRAC-WSMR.

Responsibility:
TRAC-MTRY: Review each requirement, ask for written clarification or adjustments as required and respond with a design document addressing each specific requirement. Once design is approved, develop a testing plan and schedule to support design/development/testing/transfer of models.
TRAC-WSMR: Adjust document as required. Maintain version control of document. Approve design. Develop evaluation criteria. Support testing plan and schedule as required. Request resource as needed to facilitate transfer.
TRAC-FLVN: Monitor progress of design/development/testing/transfer of models. Support as required.
TRAC-LEE: Support as required.

Usage:
Unique Identification (UID) pattern is as follows:
First number indicates a unique requirement and is usually associated with a unique issue.
Second number indicates what center is responsible for creating the requirement (1 = MTRY, 2 = WSMR, 3 = FLVN, 4 = LEE).
Third number indicates a subordinate or related requirement that pertains to requirement that is currently in existence.

A.2. SPECIFIED REQUIREMENTS

UID	Issue	Issue source	Requirement	Evaluation Criteria
1.2.0	The heterogeneous nature of the population can be significant because heterogeneous populations lead to heterogeneous type responses which can be difficult to understand. Specifically, many of the hex locations upon which wargame players focus operations contain different representative population agents. The net effect in this scenario is that while some agents might respond in a manner a player would expect, others may not.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review. Para 3.1 page 7	Population Model shall provide output that is easily understood by experienced operators/players.	This almost requires a team to play the game and say if the output is easily understood to them. I'm not sure what "right" looks like here until we play-test the game.
1.2.1	See 1.2.0	See 1.2.0	Population model shall provide output that allows experienced TRAC GS 1515 and/or FA 49 to conduct analysis.	Output loggers created to provide all data identified by the Analysis lead.
2.2.0	Aggregating population perception feedback of this type can render information less useful to the players. When aggregated, the numerical values are averaged. Averaging positive and/or negative perceptions drive the final aggregate to neutral. It also "washes out" much of the directional feedback that could otherwise be observed at lower levels. As a result of this aggregation step, the magnitude of change in the population perception as captured by the numeric output is very small. Because of the very small magnitude in change, players developed a perception that the model was not responding to their inputs. Further investigation of this issue is needed.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 3.1 page 7	Population model shall provide directional feedback that is appropriate at all levels of command (company to JTF)	Appropriate to all levels of command will vary by the location and the region. For example, the population looks significantly different to a brigade in remote areas of Afghanistan than it does in Baghdad where a brigade AO is just a section of town.
3.2.0	Survey data used to generate the initialization conditions and can be characterized a priori and outside of the CG model. This modeling approach must be evaluated for appropriateness over the course of a simulation run.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 3.1 page 9	Population Model shall provide a population representation that is consistent and reasonably understood given each time step, and the accumulation of time steps.	Same as 1.2.0 - requires a team to play the game and say if the output is easily understood to them.
3.2.1	See 3.2.0	See 3.2.0	Population Model shall include formal documentation for data requirements used to generate all initialization conditions and other model parameters.	This should be outlined in the Scenario Development Guide.
4.2.0	The wargame players currently receive the most positive (maximum) perception, the most negative (minimum) perception, and the average perception, per COIN LOE, per time step. The maximum, minimum, and average come from the aggregate population. This form of feedback in their common operating picture needs to be evaluated to ensure that players are receiving the most useful form of population perception feedback.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 3.1 page 10	The population model shall provide population feedback that is consistent with the common operating picture uniquely seen by units operating in interactive complex environments. (As described in FM 3.0 and FM 5.0)	Same as 1.2.0 This requires a team to play the game and say if the output is easily understood to them. Difficult to determine what "right" looks like without asking experienced Soldiers playing the game.

UID	Issue	Issue source	Requirement	Evaluation Criteria
5.2.0	The desired effects that come from the OAB calculation must be made clear. The Population Support overlay in the player PAVE interface tends to be one of the first, if not the only, measure that the wargame player investigates to evaluate their own success with the population in a given turn. Their understanding of the intent of this measure, along with the desired effects of this measure, must be clear to the wargame leads and to the players.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 3.2 page 11	The population model shall provide wargame players with population demographic or narrative information to give players a better understanding of the population in their area of operation.	Narratives created for each population agent.
5.2.1	See 5.2.0	See 5.2.0	The population model will distinguish the appropriate population information that will be provided to the white cell and that which will be provided each player/actor according to an appropriate level of perception (i.e. “fog of war”).	Output loggers created to provide all data identified by the White Cell lead.
6.2.0	TWG lead will review and evaluate the scenario development document, produced by TRAC-MTRY, to ensure all methodologies and processes are sufficient and appropriate for the future iterations of the wargame.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 3.3 page 13	The population model shall use data that is validated by the CODDA and/or Study sponsor.	Scenario data validated by CODDA/Study sponsor.
7.2.0	There has never been a determination of the types of problems that the TWG is capable, in its current form, to address. For example, the IW TWG is probably not a suitable venue for a material acquisition decision, unless the acquisition is expected to affect cognition or the population. Understanding the domain of problems that the TWG suits well is something that has not been done, but would be beneficial toward scoping and focusing future TWG iterations.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.2 page 15	The population model shall support (or clearly document what it cannot support) TRAC analysis to include DOTML, COA, and investment decisions.	Once a study issue is defined, there will be a measurement space meeting to define where the measurement space is. If there is not measurement space in the population model due to the type of study, it will be documented.
7.2.1	See 7.2.0	See 7.2.0	The population model will provide the level of operations that it is capable of supporting (i.e. sensitivity to actions) to include echelon and time.	Echelon/time in the study is mirrored in SIM.
8.2.0	During wargame execution, players are frustrated from the lack of desired response when attempting to improve conditions within their area of operations. Without a robust understanding of the population with which they are dealing, their frustrations are validated. In addition, in a simulation environment, there needs to be a level of intuitive response to inputs that participants can be comfortable with.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.3 page 15	The population model shall provide population starting conditions that are specific and typical for each level of command.	Starting conditions reviewed by WSMR.

UID	Issue	Issue source	Requirement	Evaluation Criteria
9.2.0	Redacting the TWG back to an UNCLASSIFIED game would reduce risk significantly and mitigate many of the integration related issues that existed amongst every model in the TWG Federation. The database could then be shared and/or made available on an UNCLASSIFIED sharepoint that model developers could access at any time and rehearse their database connection with actual data in the form that it is likely to take for the exercise. An UNCLASS wargame would also support data sharing between wargame partners.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.4 page 16	The population model shall support TRAC studies with classification of SECRET.	SIM can run either SECRET or UNCLASSIFIED.
9.2.1	See 9.2.0	See 9.2.0	All data used in support of the IW TWG shall comply to AR 25-50 (and other regulation as deemed appropriate) and contain appropriate classification and markings.	Documents are marked with classification according to AR 25-50.
10.2.0	The federation test and TWG rehearsal events this past FY became trials in connecting to the SQL server and TWG database. This type of exercise is necessary if the database is not available. However, there may be several such events scheduled and executed until the desired behavior from the models are met, and then lock the models' versions prior to the player rehearsal. Afterward, ensure that the player rehearsal is set up, effectively, as a replica of the TWG exercise to follow in the near future.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.6 page 16,17	The population model design/development personnel shall participate in all IW TWG integration events and tests.	The population model design/development personnel shall participate in all IW TWG integration events and tests.
10.2.1	See 10.2.0	See 10.2.0	The population model design/development personnel shall maintain credentials and understanding on how to access the IW TWG integration testing/staging environment.	The population model design/development personnel shall maintain credentials and understanding on how to access the IW TWG integration testing/staging environment.
11.2.0	An experimental design applied to the wargame could allow the wargame team to construct wargame vignettes that are acutely scoped and designed to observe specific wargame player behaviors and outcomes. Wargame developers would then be able to observe and share these data immediately, facilitating any need to restart a particular vignette. This would also allow for robust white cell team coordination and sharing of emerging results/insights from the wargame.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.6 page 18	The population model shall support changes to the model functionality as required to support the IW TWG 2012/13 game event.	The population model shall support changes to the model functionality as required to support the IW TWG 2012/13 game event.

UID	Issue	Issue source	Requirement	Evaluation Criteria
12.2.0	Receiving the study question late and delayed scheduling hampered development timelines. This resulted in the poor documentation and integrative testing. The timeline was a self-imposed problem that can be rectified through planning and preparation. It takes a significant amount of time to develop the scenarios for CG and Nexus that support a particular study issue. It is a measurement space problem. In addition, this game needed calibration. Conveniently, this game enabled us to differentiate between the Company Intelligence Support Team (CoIST) and non-CoIST cases, even though it has not been determined how the population models contributed to this difference. In order toadequately create a game that is sensitive to changes associated with those elements identified in measurement space, we need to have adequate calibration.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.7 page 18	The population model design/development team lead shall support all IW TWG 2012/13 planning events.	The population model design/development team lead shall support all IW TWG 2012/13 planning events.
12.2.1	See 12.2.0	See 12.2.0	The population model design/development team lead shall provide the IW TWG 2012/13 team a bi-monthly update on the progress associated with the requirement stated in this document. The update shall include resources required to address change requests and adjustments to scheduled deliveries and/or resource requirements.	
12.2.2	See12.2.0	See 12.2.0	The population model design/development team lead shall support the analysis team(s) from development of the DCMP to the completion of the documentation/final report.	
13.2.0	Following selection of the study question, MAJ Jason Whipple of WSMR traveled to Monterey to support designing the Nexus scenario. The meeting resulted in detailed description of WSMR's intent for the Nexus scenario and desired networks to be represented in Nexus. The Nexus design meeting proved very effective in the development of the Nexus scenario. Future Nexus design meetings should include discussion of all key leaders desired in the Nexus scenario, names of key leaders, and locations.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.9 page 18	The social network model shall have the same functionality that existed for the IW TWG 2011 game.	Key Leader instantiation has same functionality.

UID	Issue	Issue source	Requirement	Evaluation Criteria
13.2.1	See 13.2.0. Threat key leaders within NEXUS are associated with the Threat player's capabilities. For example, a bombmaker within NEXUS is also a bombmaker for the player. The functionality between the 2 systems needs to be improved in order to fairly represent the impacts associated with the leaders (e.g. if a bombmaker is removed from the wargame through attrition, then the removal of that person in NEXUS is adjudicated. The capabilities associated with that individual should be removed from the wargame until another assumes that role defined in NEXUS).	See 13.2.0	KLE Enhancements to CG Model Requirements Document & The social network model shall provide an appropriate impact on actor capabilities within the wargame construct for all leaders in the network and all relevant actors of the wargame. Social network(s) in SIM must be represented in terms of relationships between Key Leaders and population agents.	Key Leader component represents relationship between Key Leaders and agents.
13.2.2	See 13.2.0	13.2.1	Key Leader Representation. Key Leaders must be represented as individuals with specified characteristics and as actors within a social network(s). Must represent various personality "roles" for each Key Leader. Additionally, a clearly mapped representation of demographics, OAB, influence, and social-distance will be represented within SIM.	Key Leader component represents individuals within networks.
13.2.3	See 13.2.0	13.2.1	Key Leader Attrition. Key Leaders must be able to be attritted through kinetic and political means. SIM must account for changes within the social network when this attrition occurs.	Key Leader component allows for the removal of Key Leaders.
13.2.4	See 13.2.0	13.2.1	Key Leader Engagements. TWG players must be able to meet with key leaders via Key Leader Engagements (KLE). Players request to meet with key leaders through PAVE. KLEs could result in any of the five (5) defined outcomes: 1) messages passed via SIM events; 2)OAB Update; 3) Critical Knowledge " PAVE, Key Leader, Threat Network.	Key Leader component allows for the removal of KLEs.
14.2.0	There were still networks that WSMR needed to develop after the meetings occurring in Monterey. There was some confusion on who was in which network/location. The threat piece caused TRISA to "raise a red flag" as soon as we hit the "go" button due to their lack of participation in the planning and development process.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.10 page 19	The social network model design/development lead shall coordinate with the IW TWG lead to ensure appropriate IW Enterprise organizations are aware of changes that impact their particular expertise.	SIM functionality links up with other models through integration on DISA.
15.2.0	The players need more information on the population.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.11 page 21	The population model shall provide starting condition population data to the appropriate level necessary for players to understand the current condition and plan future operations.	Same as 1.2.0 - I'm not sure here, this almost requires a team to play the game and say if the output is easily understood to them. I'm not sure what "right" looks like here.

UID	Issue	Issue source	Requirement	Evaluation Criteria
15.2.1	See 15.2.0	See 15.2.0	The population model shall provide population response data every turn to the appropriate level necessary for players to understand the current condition and plan future operations.	Same as 1.2.0 - I'm not sure here, this almost requires a team to play the game and say if the output is easily understood to them. I'm not sure what "right" looks like here.
16.2.0	Models were changing during the actual play of the game. This created additional interoperability problems. Because there was not a table for the Operational Wrap Around (OWA), Dr. Duong was asked to "hack" in some things to the code and that made it more difficult for everyone else.	FY11 Tactical Wargame Cultural Geography Evaluation Criteria and After Action Review Para 4.10 page 19	The population model and social network model shall require no additional development once determined to be Full Mission Capable (FMC).	SIM capability is finished by date set by WSMR.
17.2.0	The impact associated with each scenario event upon the population agents is challenging for the wargame integrators and analysts to discern. Also, the appropriate distribution of the scenario events associated with possible actions adjudicated within and across the TEOs is still not well understood. This lack of understanding will impede an appropriate integration between the population models and PAVE. Also, it will heavily impact the all population measures used to inform analysis. Finally, it may mislead player decision making associated with population measures.	Post-Wargame Analysis Review	The population model design/development lead shall coordinate with the appropriate wargame lead in order to determine the set of modeling instructions that will be adjudicated within the population model.	Model Instructions are reviewed by WSMR.
17.2.1	See 17.2.0	See 17.2.0	The population model design/development lead will coordinate with the appropriate wargame lead in order to determine the appropriate number and type of modeling instructions associated with each of the intended actions within the TEO construct.	Model Instructions are reviewed by WSMR .
17.2.2	See 17.2.0	See 17.2.0	The population model shall provide documentation recording the level and type of impacts represented/adjudicated for each modeling instruction.	Model instructions tested for impact within SIM and results provided to WSMR.
CG 1.2.0	The purpose for these requirements is so that the team will be able to align the intent of each TEO with the associated impact. For example, we do not want a TEO to have too few or too many SEs. Also, if there is one main SE that represents the intended impact of the TEO, then we need to be sure and use it appropriately.		Provide the results of a DOE on the scenario events (SEs) in order to determine impact different quantities have on the population (by population stereotype).	Quantity of model instructions tested for impact within SIM and results provided to WSMR.
CG 1.2.1			Provide a scale which indicates the impact for each quantity. For example, what is the impact of adjudicating 10 SEs of the same type versus 1? Is there a threshold value where this impact is changes (i.e. bins such as: 0-5 SEs have the same small impact, 6-20 SEs have a medium impact, and so on)?	Quantity of model instructions tested for impact within SIM and results provided to WSMR.

UID	Issue	Issue source	Requirement	Evaluation Criteria
CG 2.2.0			Provide the results of a DOE on the SEs in order to determine the size of the effect of each SE on the population (by population stereotype).	Quantity of model instructions tested for impact within SIM and results provided to WSMR.

A.3. REQUIREMENTS IMPLEMENTATION

These requirements are all addressed in this technical report. Many specific testing and development requirements are in the appendices that apply to the requirement. Specifically, Appendix C, D, E, and F contain the requirements specific to development and testing. This appendix covers other requirements pertaining to coordination and participation in wargame events.

UID 5.2.0 The population model shall provide wargame players with population demographic or narrative information to give players a better understanding of the population in their area of operation

This requirement is TWG specific. Scenario developers create the population narratives with SME during the data development phase. When the team identifies a location for the next TWG, the types of population agents required will emerge. At that time, the team will need to develop population demographic or narrative information. This information should be provided to the players to heighten their understanding of the population in the wargame.

UID 5.2.1 The population model will distinguish the appropriate population information that will be provided to the while cell and that which will be provided each player/actor according to an appropriate level of perception (i.e., "fog of war")

The SIM development team created a naming structure for files that clearly delineate the files that should go to individual players for situational awareness. The team coordinated with the PAVE development team to ensure that players will receive the same, appropriate information as previous wargames. The white cell should have access to all SIM output files for analysis. TRAC-MTRY recommends putting the logged SIM files in a central location where the analysis cell can access for additional data not required for PAVE.

UID 6.2.0 The population model shall use data that is validated by the CODDA and/or Study sponsor

All data for future TWG should receive validation from the CODDA. MAJ Tom Deveans worked closely with the CODDA during the SIM 3.0 development process to produce an acceptable, validated data set for scenario development. Additionally, he developed the methodology for scenario development in use with SIM 3.0. The population model currently requires survey data from the area of interest and subject matter expert input.

Both of these will limit the resolution of the model based on the data available, as with any other model. As areas of interest for scenario settings are identified the data development process can begin. Limitations in the data should be highlighted and brought to the attention of the TWG Lead during measurement space drills.

UID 7.2.0 The population model shall support (or clearly document what it cannot support) TRAC analysis to include DOTML, COA and investment decisions

Because SIM relies heavily on specific use cases, the underlying data largely determines this range. TRAC-MTRY sees no instance where SIM cannot provide population, key leader or social network data for any TRAC endeavor. If the wargame requires information from a given population, SIM can provide insights on the reactions from the population.

UID 7.2.1 The population model will provide the level of operations that it is capable of supporting (i.e., sensitivity to actions) to include echelon and time

SIM can represent individuals or nation states. It could run for 1-day or 1000-years. However, the team recommends using SIM to represent a population at the brigade level or below (tactical level of war), and the model should run no more than 2-years time. Scenerios of 2-years or less are recommended because that demographic dimensions are static in SIM. A social dimension like age will change over time. Political affiliation is another example of a demographic dimension that might change over time. Survey data used to build the model represents a snap-shot in time and does not reflect these changes. Therefore, to run the model longer than a couple of years violates the static dimensions that model the population.

UID 8.2.0 The population model shall provide population starting conditions that are specific and typical for each level of command

The scenario file specifically establishes starting conditions. The population model will initialize the model with the parameters in that file. It is imperative that scenario developers design those worksheets with wargame starting conditions in mind. See Appendix H for the User Guide and the details of building a scenario file.

UID 9.2.0 The population model shall support TRAC studies with classification of SECRET

Scenario development for SIM will enables inputting classified data into the model. The team conducted development in an unclassified environment; however, the model can support classified studies.

UID 9.2.1 All data used in support of the IW TWG shall comply to AR 25-50 (and other regulation as deemed appropriate) and contain appropriate classification and markings

All development was unclassified; however, SIM supports the use of classified data. Once ready for a wargame the development can install the software in a classified computing environment for testing and execution. All data files will be marked with the appropriate classification level at that time.

UID 10.2.0 The population model design/development personnel shall participate in all IW TWG integration events and tests

TRAC-MTRY identified LTC Jason Caldwell and MAJ Tom Deveans to support all events. The only IW TWG integration event occurred during 17-21 September 2012.

UID 10.2.1 The population model design/development personnel shall maintain credentials and understanding on how to access the IW TWG integration testing / staging environment

The SIM development team obtained credentials and understood how to access the IW TWG integration testing/staging environment on DISA RACE. The team used the testing environment on a weekly basis during SIM 1.0 and 2.0 testing. By May 2012 the only updated versions on DISA RACE were SIM and PAVE so it became easier to email the versions back and forth to conduct integration testing.

UID 11.2.0 The population model shall support changes to the model functionality as required to support the IW TWG 2012/13 game event

SIM is adaptable to changes required of the IW TWG. This is accomplished through the object oriented design of the model. The goal of all development will be a loose coupling so that modules can be removed, modified and refactored into the model. However, there are several modules that are absolutely necessary to SIM (See UID 3.2.0). See Appendix C and Appendix E for overviews of the design. The JavaDocs and event graphs provide a more detailed look.

UID 12.2.0 The population model design/development team lead shall support all IW TWG12/13 planning events

TRAC-MTRY identified MAJ Tom Deveans to support all TWG 12/13 planning events after transition of the model.

UID 12.2.1 The population model design/development team lead shall provide the IW TWG 2012/13 team a bi-monthly update on the progress associated with the requirement stated in this document

The SIM Development Team conducted three (3) In Progress Reviews (IPR). One in February, beginning the transition process. Another in June after competing SIM 2.0, and the final IPR in September. After IPR#1, the IW TWG Lead stated that direct coordination for transition would happen between Ms. Clark and LTC Caldwell. During Ms. Clark's training in Monterey, weekly updates to the IW TWG Lead occurred. Upon her return to TRAC-WSMR in May 2012 weekly updates continued between LTC Caldwell and Ms. Clark. Ms. Clark briefed the IW TWG Lead on a regular basis between the IPR.

The updates to Ms. Clark included:

- Current status of development.
- Questions specific to the use case for the IW TWG.
- Current status of documentation.
- Status of contracted support.
- Technical support resources required post-transition.
- Requests and adjustments to scheduled deliveries.

UID 12.2.2 The population model design/development team lead shall support the analysis team(s) from development of the DCMP to the completion of the documentation/final report

TRAC-MTRY will support the analysis team(s) in the next TWG from development of the Data Collection Management Plan (DCMP) to the completion of the documentation/final report.

UID 15.2.0 The population model shall provide starting condition population data to the appropriate level necessary for players to understand the current condition and plan future operations

There was not a wargame to tailor data for in 2012. The next IW TWG scenario development team should provide the cultural narratives used to develop scenarios. These narratives should be distributed to TWG players to understand the types of population agents in their operational environments. Furthermore, the scenario development team should prepare a player primer on the distribution of population agents by hex (or other geographic marker) to increase a players' situational awareness. TRAC-MTRY is prepared to support this effort as necessary.

UID 15.2.1 The population model shall provide population response data every turn to the appropriate level necessary for players to understand the current condition and plan future operations

SIM provides population response data every turn. SIM outputs aggregated population responses by stereotype by area (hex). Loggers enable the segregation of individual agent responses for clarity. Review the testing data to see how SIM can isolate specific agents. PAVE needs to determine how to display this to a player.

UID 16.2.0 The population model and social network model shall require no additional development once determined to be Full Mission Capable (FMC)

Upon final delivery SIM will require no additional development for defined use cases. At a minimum, one use case will be the next TWG.

UID 17.2.0 The population model design/development lead shall coordinate with the appropriate wargame lead in order to determine the set of modeling instructions that will be adjudicated within the population model

The development team coordinated with the wargame leads to determine the set of inputs to adjudicate within the population model. Coordination included determining the outputs needed to display impacts to the players and the white cell. SIM interfaces with PAVE; therefore, LTC Caldwell coordinated with PAVE developers to ensure inputs and outputs from the model comply with all integration standards. Because there was no TWG in 2012, the team used the same set of model instructions from TWG 2011 for testing.

APPENDIX B. TRAC-MTRY AFTER ACTION REVIEW (AAR) FOR THE 2011 IW TWG

The purpose of the TRAC-MTRY after action review (AAR) was to capture lessons learned from the FY11 IW TWG. The lessons learned informed the continued development of models and tools used for TWG support and highlighted issues for attention during the transition of SIM during FY12. See Appendix B for the complete AAR. This appendix follows the Observation, Discussion, and Recommendation format. Attendees at the AAR were:

- LTC Alt, Director TRAC-MTRY.
- Mr. Jackson, Deputy Director TRAC-MTRY.
- MAJ Evangelista, TWG 2011 Analysis Cell.
- CPT Brown, Scenario Developer for CG Model.
- MAJ Vargas, TWG Player (D Co).
- Mr. Pearman, TWG Analysis Cell.
- Dr. Duong, Nexus Modeler.
- Mr. Ruck, CG Modeler and designer of KLE component of SIM.
- Mr. Yamauchi, CG Modeler and primary developer of SIM.
- MAJ Deveans, Analysis Cell for TWG 12.
- MAJ(P) Caldwell, Models Cell for TWG 12 and SIM Transition Lead for FY12.

B.1. OBSERVATIONS

B.1.1. Cultural Geography (CG) Database Interaction

Observation. Scenario events did not process in the Cultural Geography Model (CG).
Discussion. The scenario events produced from the wargame were not processing in CG. As a result, the two TWG events turned into an exercise in connecting to the database.

Classification issues create challenges for development and testing. Being able to work with actual data on the Non-secure Internet Protocol Router (NIPR) network would facilitate faster development and testing. There will be an effort to declassify the database, or at least, remove classified data from the database so it can be distributed on NIPR. Of note, CG only requires about a half dozen tables from the Players Adjudication Visualization

Environment (PAVE). Having these specific tables unclassified for development and testing would be useful.

The challenge is that there may or may not be a method to identify and trace the classified data that was used to develop the probabilities. It might be unknown which probabilities were based on the classified vs. unclassified data. Culturally, dumping all information into one classification and below without any traceability is a bad practice. For example, improvised explosive device (IED) data populated from the Tactical Ground Reporting Network (TIGR) creates an increased classification level. It will take a little work to bring from the classified side to the unclassified side. Two benefits of declassifying the database would be analysis and cost. Analysis could occur begin sooner on an unclassified database, and an unclassified, distributed database would reduce associated travel costs.

Recommendations.

- Have an event that replicates what occurred at the TWG as a full dress rehearsal prior to TWG execution.
- Replicate the capability at White Sands Missile Range (WSMR) here in Monterey. For example, set up the entire functionality in the STBL to include PAVE, the SQL database, and CG.
- Develop on Non-secure Internet Protocol Router Net (NIPR) and move to Secure Internet Protocol Router Net (SIPR).

B.1.2. Documentation

Observation. There was a lack of documentation for the event and the models.

Discussion. There was a lack of information on the way the integration was intended to happen. Last year (2010), Dr. Duong made a document on how PAVE and Nexus would interact. This year (2011), she thought it was the responsibility of the database personnel; however, the documentation was not at the level necessary to specify how different pieces of data are supposed to align. Because that documentation did not exist and there were only occasional emails back and forth assuming that the other parties knew what they meant, this alignment did not occur. Most of the problems in the game, including the restart, were a result of not having any integrated testing.

The integrated testing must occur in two phases. First, there is documentation and getting the software ready phase, and second, there is an execution phase where all software ran together. This could have all been done on NIPR passing the database back and forth. It is unrealistic to expect the little time together in WSMR will produce “clean” software that will work as expected.

This is also a problem with CG. For example, there are a lot of fields with no documentation other than the data type. In 2010, all data was written to tables, and in

2011, the data was sent to the database. This is the first year for the employment of this technique.

Recommendation.

- Produce a Word document that explains the field descriptions and type.
- Freeze the development of database to allow modelers to complete integration prior to the wargame.

B.1.3. Execution of the Wargame as an Experiment

Observation. The current construct of the TWG makes conducting analysis of decision making and detecting the measures that address the primary issue for analysis very difficult. A thoughtful experimental design applied to the TWG might eliminate much of that difficulty.

Discussion. An experimental design applied to the wargame could allow the wargame team to construct wargame vignettes that are acutely scoped and designed to observe specific wargame player behaviors and outcomes. Wargame developers would then be able to observe and share these data immediately, facilitating any need to restart a particular vignette. This would also allow for robust white cell team coordination and sharing of emerging results and insights from the wargame.

Recommendation.

- Revisit the construct of the wargame and potentially break up into discreet vignettes in order to facilitate traceability, analysis, and data management.
- Recommend to Mr. Solis and Mr. Gard the utilization of scenario/vignette experimental construct.
- This could also include post-game analysis to have experimental analysis of the vignettes using decision points and statistics from the game.

B.1.4. Schedule

Observation. The team received the study issue too late in the process. The schedule for future TWG needs to be established early.

Discussion. Receiving the study question late combined with delayed scheduling of wargame events hampered development timelines. This resulted in the poor documentation and integrative testing (paragraph a2 of this appendix). The timeline was a self-imposed problem that can be rectified through planning and preparation. It takes a significant amount of time to develop the scenarios for CG and Nexus that support a particular study issue. It is a measurement space problem. In addition, this game needed calibration. Conveniently, this game enabled us to differentiate between the Company Intelligence Support Team (CoIST) and non-CoIST cases. In order to adequately create a game that is able to differentiate them, we need to have adequate calibration.

Recommendation.

- TWG needs calibration.
- The study question should be determined much sooner.
- Look at cycle length of the TWG. Perhaps an 18-month vs. 12-month cycle makes more sense.

B.1.5. WSMR trips to MTRY

Observation. WSMR personnel traveling to Monterey had a positive impact on the TWG.

Discussion. Once we did have the study question, MAJ Jason Whipple traveled to Monterey to design the Nexus scenario and engage in discussions led directly to successes in the TWG. In the past, we had an elaborate Task, Event, Outcome (TEO) discussion to conduct the CG and Nexus events, but this time, we had a separate Nexus discussion which was helpful.

Recommendation.

- Sustain WSMR trips to Monterey next year.

B.1.6. Key Leader and Intelligence Involvement

Observation. WSMR takes ownership for managing the networks and TRADOC Intelligence Support Activity (TRISA) involvement.

Discussion. There were still networks that WSMR needed to develop after the meetings occurring in Monterey. There was some confusion on who was in which network/location. We were not able to get WSMR engaged on some issues modeled in CG (e.g. Infrastructure). The threat piece caused TRISA to “raise a red flag” as soon as we hit the “go” button due to their lack of participation in the planning and development process.

Recommendation.

- Kristen Clark extended stay here in Monterey to go through CG and Nexus scenario development. This should improve the connection to TRISA early. Additionally is will create "ownership" instead of "buy-in". Remember, the goal for next year is that it is a WSMR project with consulting from MTRY.

B.1.7. Population Guides

Observation. The players need more information on the population.

Discussion. This has not been done in the past, but was mentioned as a future requirement. As WSMR takes ownership of SIM, they should provide succinct descriptions of the population in the models for the player's benefit. The TWG kickoff was delayed at the very beginning in order to adjust the population scenario to reflect an overall higher attitude toward the Taliban. This action was in direct contradiction to the detailed data development conducted with regard to each wargame player's starting OAB related conditions, to include the threat conditions. All players that have the appropriate level of expertise regarding population relations, distributions, or attitudes toward any representative actor within the conflict environment must be involved in the cultural scenario development process early as their input in ensuring the accuracy and synchronization of the cultural scenario is in keeping with the overarching IW TWG scenario is critical.

Recommendation.

- Use Department of Defense (DoD) guides on population. The atmospherics guides produced by the Marine Corps Intelligence Activity out of Quantico can serve as a model.
- TRAC-MTRY hosts/leads a cultural scenario development workshop, either independently or as part of a pre-existing workshop, in which the data development methodology is described in detail and roles toward further development are clearly identified. Scenario development representatives from TRAC-WSMR, TRAC-FLVN, TRISA, and any other appropriate center involved in data and scenario development should be in attendance.

B.1.8. CG and the Inputs that Players Need

Observation. CG is not providing the information that players need during the game.

Discussion. There is a need to determine the essential pieces of information that a player needs. The purpose of CG is to provide the stimulus for the players to respond and behave in a realistic way. There is a need to go back to the basics that support this purpose.

From a population modeling standpoint, the population was randomly distributed across the map. We were supposed to use ERDC geographer out of the University of Illinois to distribute the population in a representative manner. There was a plan to distribute the population effectively; however, this did not occur. The plan received was not usable due to inadequate guidance; he didn't follow the guidance provided and used poor data sources. In terms of observed attitudes and behaviors, it is difficult to understand what this means in a manner that makes sense to a human player. There are five probabilities and there is an aggregation scheme that was fixed (but may not be adequately fixed). There is movement, but the movement is very slight. Perhaps the model should run "hotter" to exaggerate for training effect. The alternative to running hotter is to set expectations and focus more on relationships. Relationships between players and the game should be more persistent. For example, when you go into a certain neighborhood, you should encounter the same people. Even if the changes are slower, there is a repeated engagement that occurs.

The way we derive the theory of change is based on the way polling data changes over time and how the subject matter experts (SME) estimate that the population will respond. From a player standpoint, if you don't see change, there is no reason to change your course of action (COA). If there is no difference in the way a population will respond, a player sees no utility in changing his COA. Alternatively, if a player is using a good COA but does not see results, they may unwisely change their COA because it appears to have no positive effect on the population.

Recommendation.

- Provide adequate guidance for the distribution of the population.
- Do a better job of setting the expectations for the model.
- Make the relationships between the players and the population more persistent.
- This issue requires a working group to explore methods for improving the models in greater detail.

B.1.9. Development During the Game

Observation. Development on models occurred during the execution of the TWG.

Discussion. Models were changing during the actual play of the game. This created additional interoperability problems. Because there was not a table for the Operational Wrap Around (OWA), Dr. Duong was asked to "hack" in some things to the code and that made it more difficult for everyone else. Mr. Gard and Mr. Solis have indicated that we will not do model development during the execution of the next TWG. Mr. Gard has a three tier approach to models. Tier three is during the execution and no development occurs in that tier. There will be a "freeze" on development at a predetermined point. Mr. Works did not completely agree with this policy. He believes that the modelers are adept enough to make small changes during execution.

Recommendation.

- Need to make the decision to allow/disallow model development before the TWG.
- If development is allowed during game play, then it needs to be limited.

B.1.10. Player Perception of Reality

Observation. There was no discussion of what reality looks like with the players.

Discussion. There is a need to have a discussion with all the wargame partners involved about what the population responses should look like. Similar to the issue with player input needs (section a8 of this appendix), there might also be a need to exaggerate reality to assist player perception of population responses. We already have TEO related to CG. They exist as part of the outcomes. You could potentially associate population responses to these outcomes. How this would feed into PAVE needs to be explored as a possible improvement for player perception. This all comes back to planning to determine solutions. For example, a lookup table might be a simplest solution. Also, scaling percentages of change is another possible method. Mr. Gard, Mr. Solis, and Dr. Lambert all expressed the desire to have more traceable causality back to the population.

Recommendation.

- Meet with the players in order to outline what reality (game outcomes) look like, when certain conditions exist.
- Plan and define the tools needed to meet the intended outcomes.
- Meet ahead of time to determine expected outcomes (simpler is better).

B.1.11. Determine Causal Relationships

Observation. There is a need to trace causality in the population.

Discussion. Mr. Gard, Mr. Solis, and Dr. Lambert all expressed the desire to have more traceable causality back to the population. CG set up that effects go 20km (half a BN AO). Actions potentially affect entire AO as a result. This is not general to all entities, but it does occur in many cases.

Recommendation.

- Have a benchmark scenario ("Hello World") where events are traceable all the way through.
- Have the models defined with theories of cause and effect.
- Build simple conceptual models that represent the underlying mechanisms that are in the agent models. Present these to the white cell analyst team, so they have expectations as to the theories and cause and effects that are inherent in the models.

B.1.12. Database

Observation. The database is critical to the success of TWG.

Discussion. We cannot underestimate the value of an online database for future development. We need more of what the models dumped into the database. This supports development of models and an analysis plan. So while the database was a challenge, it was a tremendous enabler for analysis. It has all sorts of good features allowing rehearsal of the analysis and diagnosis of what is going wrong.

Recommendation.

- Establish an online database as soon as possible. This should be sustained and expanded so analysts can access more data/info from CG and Nexus.
- Create a method for business intelligence processes, online analytical processing, and diagnostics. Set of gauges would be helpful.

B.1.13. Knowledge Elicitation Process

Observation. Knowledge elicitation process needs improvement.

Discussion. Relook how we get this information through planning, education and training, and proven methods. How we get information from the players needs to improve. Once they fill out the surveys once or twice, the survey provides limited utility. The interviews need more structure. They should have interviews and focus groups. They should apply cognitive science to how they approach the process.

Recommendation.

- Need to improve the way we elicit knowledge from the players.
- Conduct a cognitive task analysis (CTA) workshop to train some analysts.
- Leverage expertise from everyone throughout the process.

B.1.14. Analysis Plan

Observation. The analysis plan needs improvement.

Discussion. The analysis plan was too broad when received and lacked continuity. Analysis team felt that they rigidly had to adhere to the developed plan. Anything we can do to scope the analysis requirements. Ensure we completely think through the data that can be collected and measured " half of the measures were incomplete. Ensure we maintain continuity. There is definitely a gap in the analysis plan and data collection effort. What was being collected from the surveys and interview process demonstrated little connection between the plans. In fact, there was some backwards effort trying to map things collected back to the plan instead of the other way around.

Recommendation.

- Think through the data that needs to be collected and measured.
- Maintain continuity.
- Rehearse the analysis " not just for the players and modelers.

B.2. CONCLUSIONS

This AAR captures lessons learned from the FY11 Tactical Wargame (TWG) in order to assist in the continued development of models and tools used in support of the TWG and Social Impacts Model (SIM) during FY12. As a result of this AAR, an additional working group will convene to determine recommendations for improving the player inputs needed from CG (Observation A8). Lessons learned will guide immediate fixes to SIM 1.0, and they shall be adhered to for future development of the SIM.

APPENDIX C. SIM 1.0 TECHNICAL DESIGN

C.1. SPECIFIED REQUIREMENTS

TRAC-WSMR specified the following requirements for SIM. This section does not address all of the requirements, but rather, the requirements met by SIM 1.0. See Appendix A for a complete list of requirements for SIM Transition.

UID 1.2.1 Population model shall provide output that allows experienced TRAC GS1515 or FA49 to conduct analysis

Loggers in SIM determine what output come from the model. Under the final IW TWG 2011 configuration, SIM 1.0 produced 33 comma separated value (CSV) files. At a minimum, attitudes, issues, and infrastructure status have output files. These output files contain the same data output to PAVE, and they were the source for the testing done on all versions of SIM. The files are easily analyzed using excel or other more robust statistical software. The following lists outline the minimum standard for output files that provide population responses.

Attitudes There shall be one attitude CSV file per actor. At a minimum, each CSV file contains a column for:

- Replication: Simulation replication.
- Time: Simulation time step.
- Logger Name: Name of the logger for the data element. This serves as a check that all data is from the proper logger. The code filters data sent to this file based on this logger name.
- Entity Element: Agent, actor or infrastructure.
- Entity Name: Name of the stereotype and the instance identifier.
- Location: Hex or other spatial descriptor for game location.
- Property Name: The property name of the OAB, for example.
- Attitude: Output variable to describe the action of the attitude (e.g. OAB_TowardsANA)
- Negative Active (NA): Probability of having a negative active OAB towards the object of the attitude.
- Negative Passive (NP): Probability of having a negative passive OAB towards the object of the attitude.

- Neutral (N): Probability of having a neutral OAB towards the object of the attitude.
- Positive Passive (PP): Probability of having a negative active OAB towards the object of the attitude.
- Positive Active (PA): Probability of having a negative active OAB towards the object of the attitude.

Issues There shall be one CSV file per issue. At a minimum, each issue CSV file contains an output column for:

- Replication: Simulation replication.
- Time: Simulation time step.
- Logger Name: Name of the logger for the data element. This serves as a check that all data is from the proper logger. The code filters data sent to this file based on this logger name.
- Entity Element: Agent, actor or infrastructure.
- Entity Name: Name of the stereotype and instance identifier.
- Location: Hex or other spatial descriptor for game location.
- Property Name: The property name of the issue.
- Adequate: Probability of the entity having an issue stance of “adequate” for a given issue.
- Inadequate: Probability of the entity having an issue stance of “inadequate” for a given issue.

Infrastructure There are several output files for analysis related to infrastructure in SIM. First, there is a CSV file for the number of agents arriving at infrastructure nodes. At a minimum, the CSV file for infrastructure arrivals contains an output column for:

- Replication: Simulation replication.
- Time: Simulation time step.
- Logger Name: Name of the logger for the data element. This serves as a check that all data is from the proper logger. The code filters data sent to this file based on this logger name.
- Entity Element: Agent, actor, or infrastructure.

- Entity Name: Name of the infrastructure node composed of simulated infrastructure name and location (e.g. InfraElectricity_AA171).
- Observations: How many observations in this time period.
- Low: Low number of arrivals in the time period.
- High: High number of arrivals in the time period.

Similar to the infrastructure arrival CSV, the following CSV files will show outputs from SIM for analysis using the field (column) structure above:

- Serviced: number of agents serviced by a specific infrastructure node.
- Balked: number of agents who arrived for service, but decided to leave before being served by a specific infrastructure node.
- Queue Size: describes the length of the queues for service at each specific infrastructure node.

These loggers provide the ability to conduct many standard operations research (OR) analysis processes. Those processes include, but are not limited to:

- Average time in the queue.
- Average queue length.
- Server utilization.

UID 3.2.0 Population Model shall provide a population representation that is consistent and reasonably understood

Dr. Arnold Buss of the Naval Postgraduate School teaches SimKit Modeling. He espouses that event graphs are the foundation, or design document, for any SimKit model. From its inception in 2008, the Cultural Geography model relied on event graphs as the foundational design document. The following event graph figures outline the population representation in SIM 1.0. A brief explanation of each event graph follows the components. JavaDocs references provide additional clarity.

Figure C.1 outlines the basic architecture for an agent in SIM 1.0. Agents are objects. They have attributes (fields) and behaviors (methods) defined in the Java programming language. As it applies to the code, attributes refer to the variables and constants that define each instance of an agent and the agent's state. An object-oriented programming way to say this is: "an instantiation of an agent". In discrete event simulation (DES), these variables are known as "state variables"[1]. Each instantiation can have different values for

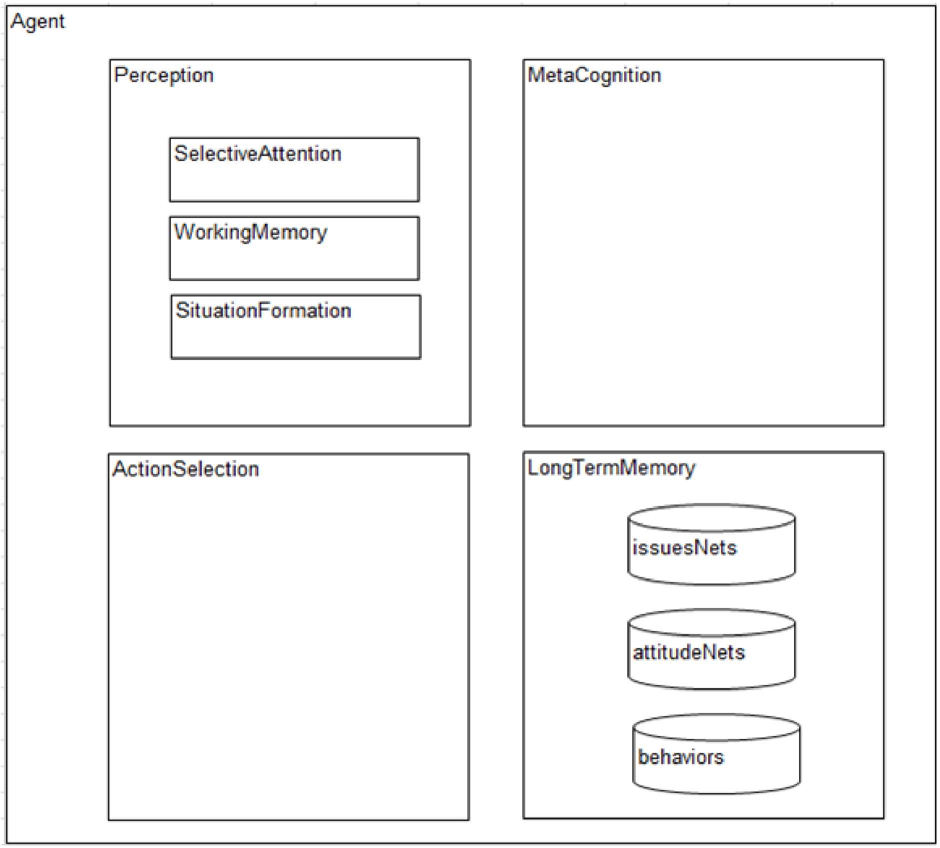

Figure C.1: Agent's Cognitive Architecture in SIM

state variables associated with the agent. In SIM, a stereotype has identical values for state variables and constants upon instantiation. Once the model runs, different instances of the same stereotype most likely have different attributes due to the different events (simulated experiences) of each agent.

Similarly, behaviors refer to the methods (or functions) that manipulate an object in Java. For an agent, there are numerous behaviors, not to be confused with population behaviors. For the sake of clarity, this list of actions will be referred to as "methods" as they relate to objects. A few methods for an agent are to:

- Do a behavioral action.
- Communicate with other agents.
- Consume a resource.
- Receive a resource for consumption.
- Forward percepts to their cognitive architecture as they arrive.

In the simplest terms for SIM, agents can perceive their environment, seek resources, and communicate with others. Figure C.2 demonstrates the major processes necessary for an agent to perceive their environment. First, an event occurs which creates a Percept object. A percept is an object instantiated for each agent that perceives the event. For example, if a "CFConductsCheckpoint" scenario event occurs,one or more agents in that hex will receive a percept object for that event. The agent will forward that percept to their cognitive architecture for processing. The ultimate result could be a myriad of outcomes; however, the following list captures a few possible results:

- Update OAB.
- Update Issue Stance.
- Communicate.
- Do nothing.

Many of these outcomes could all occur as a result of one perceived event by an agent in SIM.

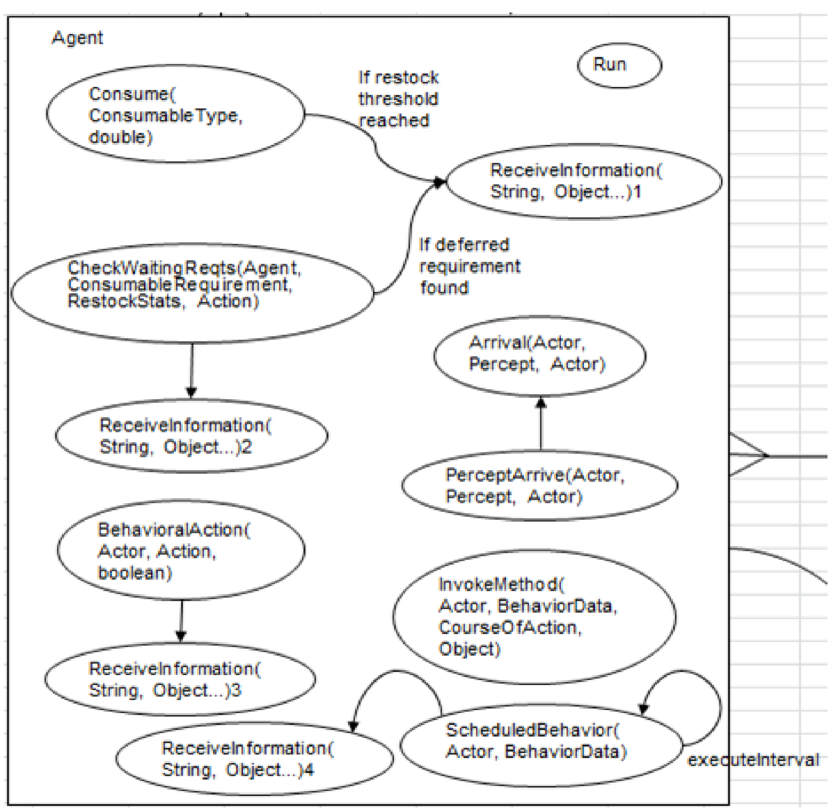

Figure C.2: Agent Component

Figure C.2 visually depicts what an agent does. First, focus on the "ReceiveInformation" node toward the top-right of the event graph. As described, the receive information node takes an agent by name (string) and the action to be processed (object). The Percept Umpire module is "listening" for the receipt of information. The event graph represents the listening action withthe symbol that looks like a crow's foot (there are two of them between the two components in Figure C.3). The listener attaches from the Agent module to the PerceptUmpire component. Think of this crow's foot like a stethoscope, listening for a specific event - an arrival at the ReceiveInformation node.

Notice that ReceiveInformation appears in both components. This is standard SimKit structure. The ReceiveInformation node in the Agent component is known as the "Source", and the ReceiveInformation node in the PerceptUmpire component is the "Listener". This notation makes the event graphs easy to follow when multiple components connect together.

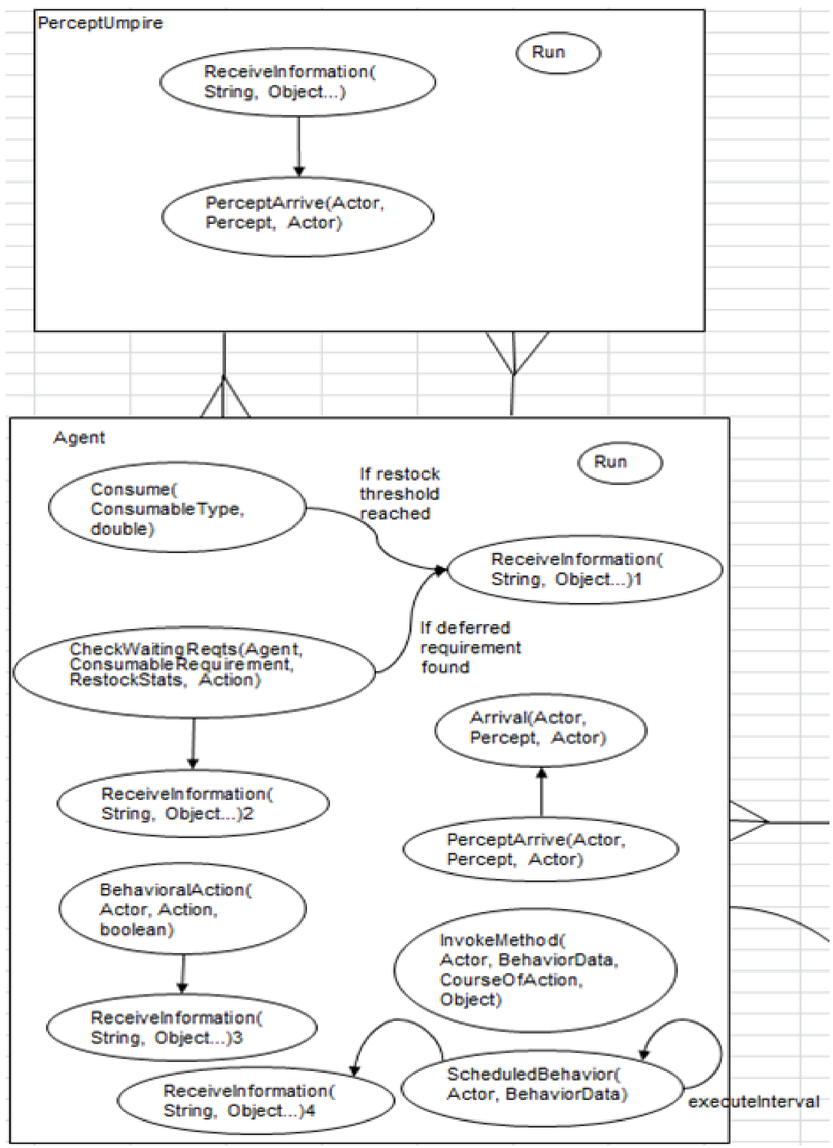

Figure C.3: Percept Umpire and Agent Component Listening Structure

Once the information comes to the Percept Umpire, it is immediately forwarded to the PerceptArrive node. The PerceptArrive node takes three parameters that enable the cognitive architecture to calculate and attribute effects in the model:

- The entity (agent) who receives the Percept object.
- The Percept for processing.
- The entity (actor) sending a Percept to the entity receiving the Percept.

Figure C.4 shows this activity.

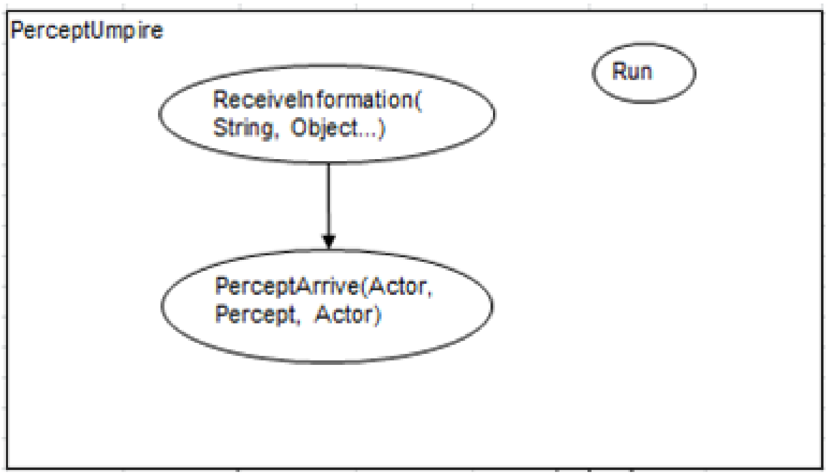

Figure C.4: Percept Umpire Component

The sending entity may be null if the action is an event. If the action is a communication event, the sending entity will be the agent that is initiating the communicaiton. Using the previous example where Coalition Forces conducted the check point, the sending entity would be null even though the event is associated with the CF actor. The scenario file defines the effects attributed to the actors intiating an action. The Percept is the presence patrol event. The receiver of the percept is any agent in the hex where that patrol happened and the Percept Umpire will adjudicate which agents "see" the activity. These three pieces of information trigger the arrival of the Percept in the Perception component. The Agent component is "listening" for the arrival of a Percept via the PerceptArrive node (Figure C.2).

The Perception Component is "listening" to the Agent Component for an Arrival event. The Perception Component is the first of the four major components in the cognitive architecture (Figure C.1) within SIM. Perception has three sub-components:

- Selective Attention (Figure C.6)
- Working Memory

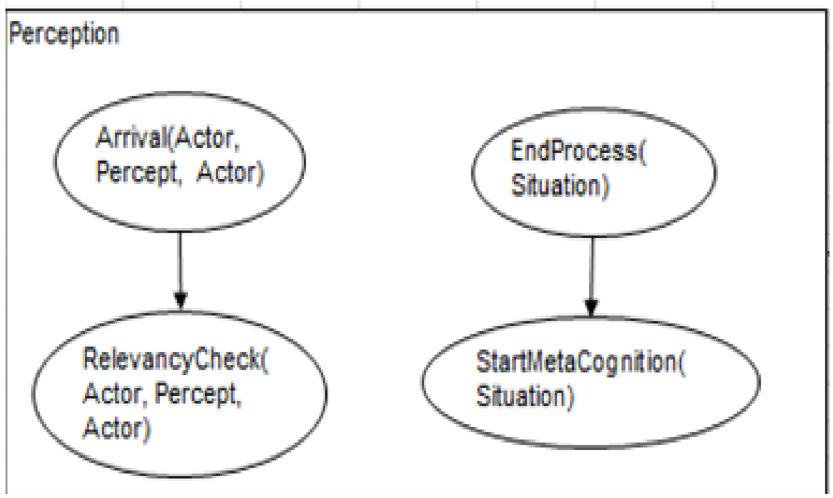

Figure C.5: Perception Component

- Situation Formation

After the arrival of a percept object to the Perception component in SIM, SIM does a relevancy check. The Selective Attention Component is "listening" for a relevancy check. The relevancy check determines if the percept should be added to the Working Memory of a specified receiving entity. Figure C.6 outlines the relevancy check done by the Selective Attention Component. The words between nodes (to the right of the arrow) indicate a conditional statement. In this case, if the Percept is relevant to an agent, then the relevant Percept continues along the scheduling edge (the arrow-line) in the Perception Component with the same three parameters mentioned earlier (receiver, the Percept, and sender of the Percept). Scenario designers determine relevancy in the scenario file by setting how long it is until a Percept is "stale" (Column A of the Cognitive Architecture Worksheet in the Scenario File).

The Working Memory Component (Figure C.7) "listens" for a relevant Percept. When it "hears" the Selective Attention Component "fire", that relevant percept arrives in working memory. Another conditional statement determines if there is room in working memory (WM) for the Percept. Of note, the scenario file sets the capacity of working memory (Column B of the Cognitive Architecture Worksheet in the Scenario File).

Once working memory reaches capacity, the Percepts are sent to the Situation Formation Component (Figure C.8) via the ProcessCurrentSituation node. The ProcessCurrentSituation node's parameter is a Java List containing the data type WorkingMemoryEntry. WorkingMemoryEntry is an interface, providing a contract (or set of rules) for extracting information from a Percept object. Specifically, WorkingMemoryEntry has methods to get a specific Percept, the receiver of the Percept, and the sender of a Percept. The SituationFormation Component contains its own ProcessCurrentSituation node that is "listening" for Working Memory to send it the list of WorkingMemoryEntry to process.

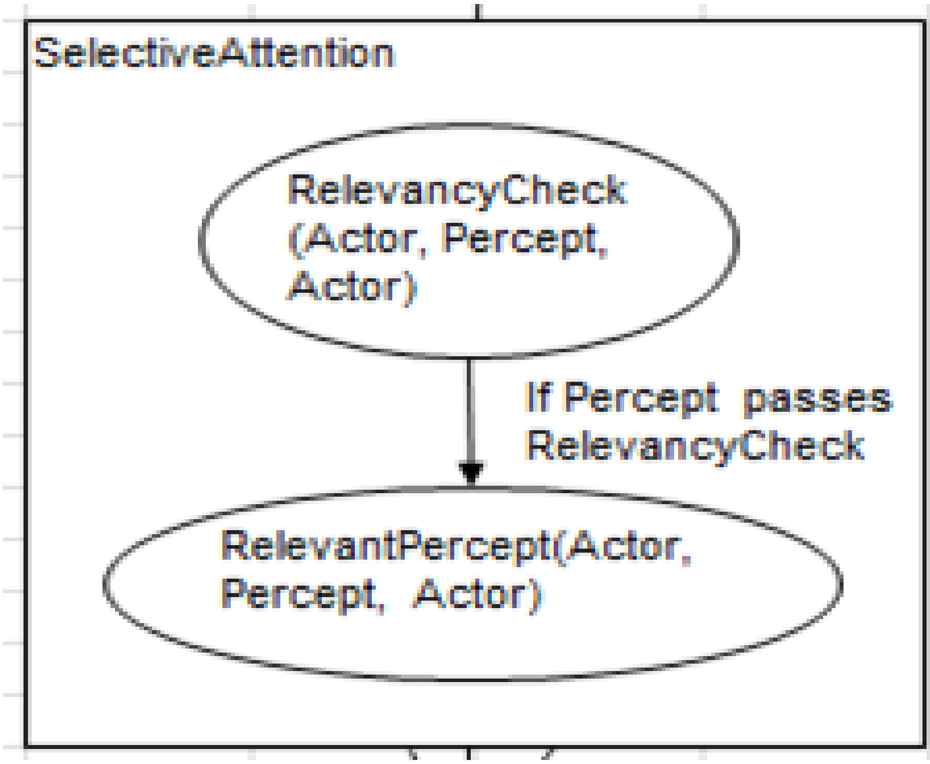

Figure C.6: Selective Attention Component

The Situation Formation Component forms a situation based on the percepts held in the specified list of WorkingMemoryEntry. At the conclusion of processing objects in working memory, an EndProcess method occurs. The Perception Component (Figure C.5 & C.9) is "listening" for this end. The Situation Formation Component returns a "Situation" back to the Perception Component. A Situation is another Java Interface that consists of all the Percepts in working memory at a given point in time. A Situation forms when the working memory reaches capacity and is transferred to the Situation Formation Component. The Situation Interface has methods to get a list of all Issue Percepts and all Opportunity Percepts in a specific Situation.

The Perception Component schedules a StartMetaCognition event. The StartMetaCognition node takes a Situation as its parameter. The Metacognition Component (Figure C.10) "listens" for a StartMetaCognition event. This begins MetaCognition, the second of the four components the cognitive architecture for each agent (Figure C.1). The StartMetaCognition node in the MetaCognition component takes the specified Situation formed in the Perception component to start the development of situation understanding. Next, the UpdateLongTermMemory node gets scheduled with the Situation as its only parameter. This node is the source for the listener UpdateLongTermMemory node in the LongTermMemory component.

The LongTermMemory Component (Figure C.11) commits the Situation to "memory" and updates issue stances and OAB. SIM does this via the SelectAttitudePosition node for

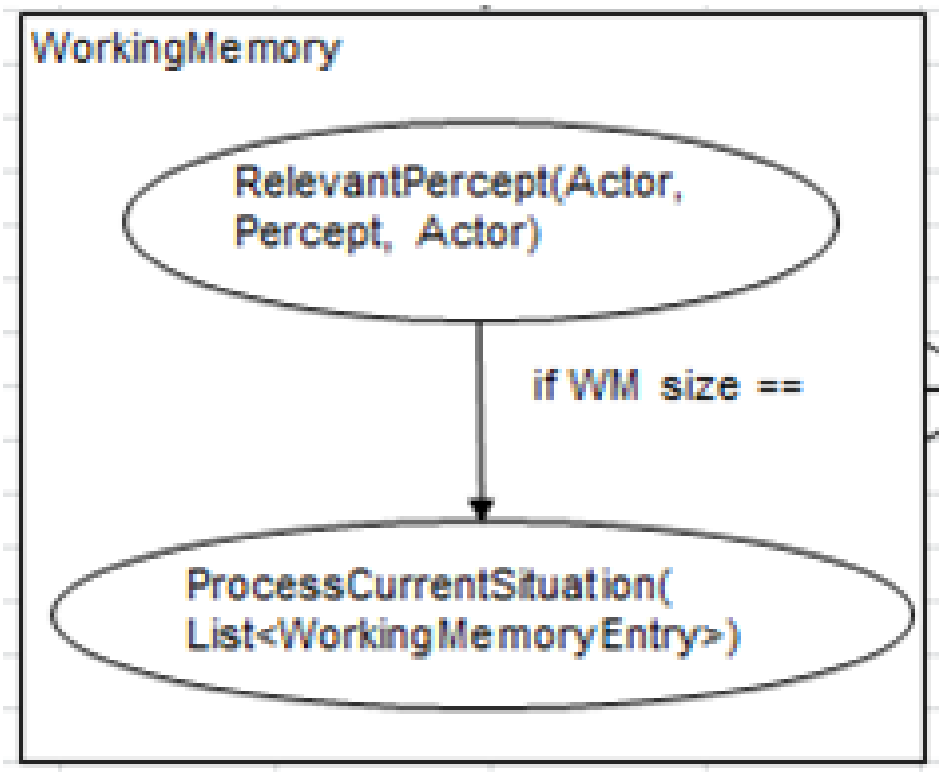

Figure C.7: Working Memory Component

updating OAB. If an attitude BBN exists related to the Percepts in the Situation arriving to long term memory, an update occurs and results are stored for output and later retreival.

If an agent has at least one behavior to conduct, the MetaCognition component schedules an UpdateMotivation event. The UpdateMotivation node determines what action an agent is most motivated to take based on motivation scores (See JavaDocs for specific documentation). Those actions might be:

- Communicate.
- Seek Resource.
- Do Nothing.

Based on the motivation scores, an agent forms expectations, sets goals, determines the method for achieving the motivated behavior, and then schedules an IdentifyAction event. The IdentifyAction node in the MetaCognition Component is the source for the listening IdentifyActions node in the ActionSelection Componenet(Figure C.12).

The ActionSelection Component first determines the decision method used by an agent. SIM 1.0 contains the ability to choose between RPD or EL. Mental Simulation is "stubbed" out in the code for future development; however, it is not a functional option.

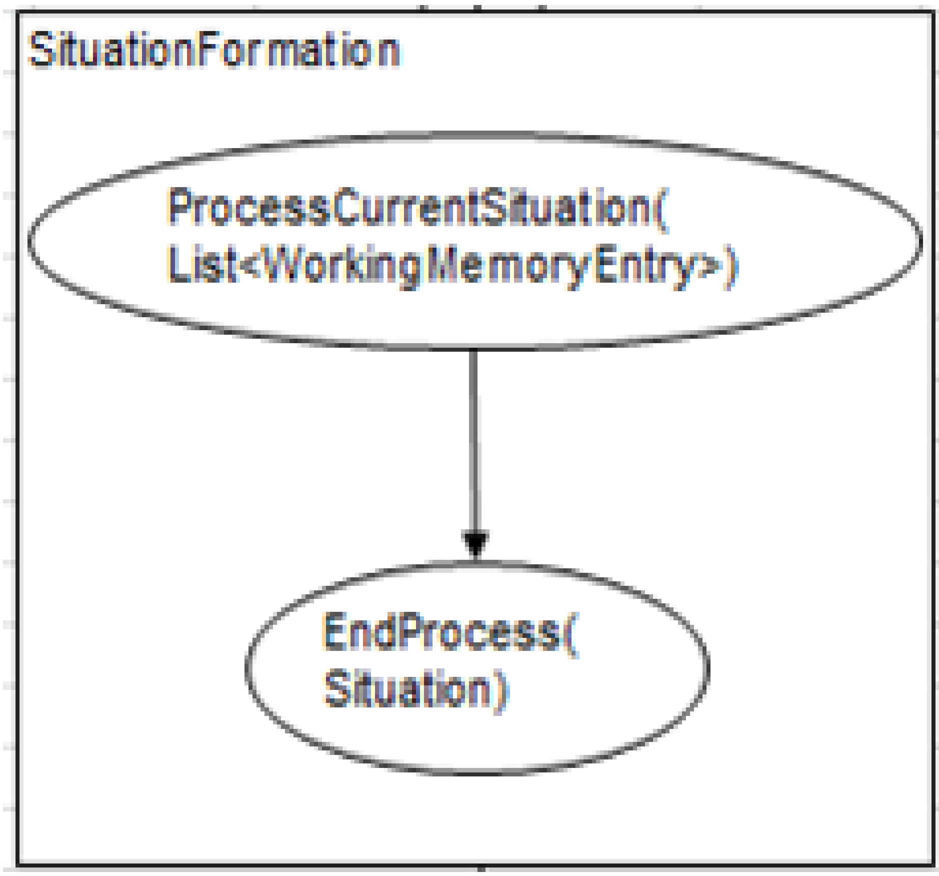

Figure C.8: Situation Formation Component

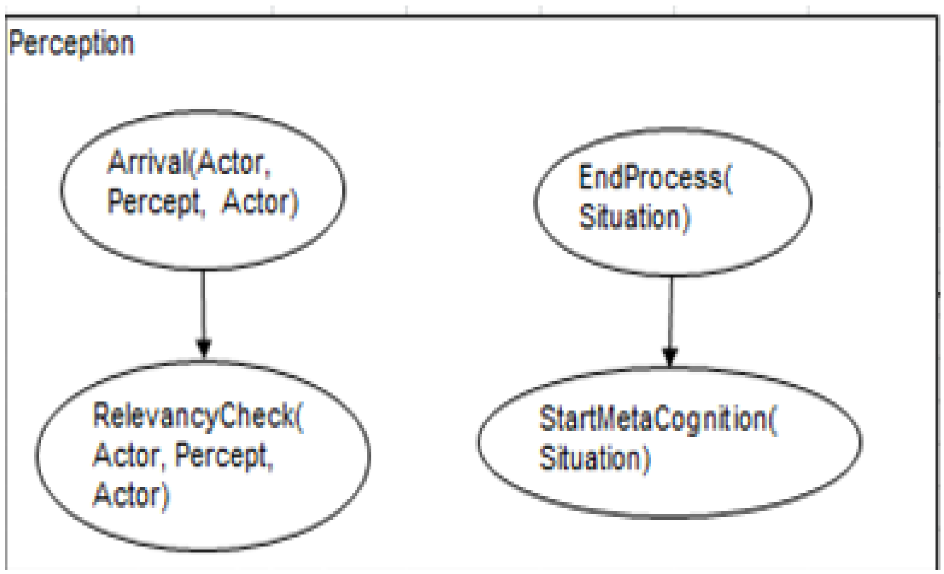

Figure C.9: Perception, listening for the end of Situation Formation Component.

The ActionSelection Component will choose RPD for an agent when the agent has performed an action a specified number of times. This experience threshold is in Column E

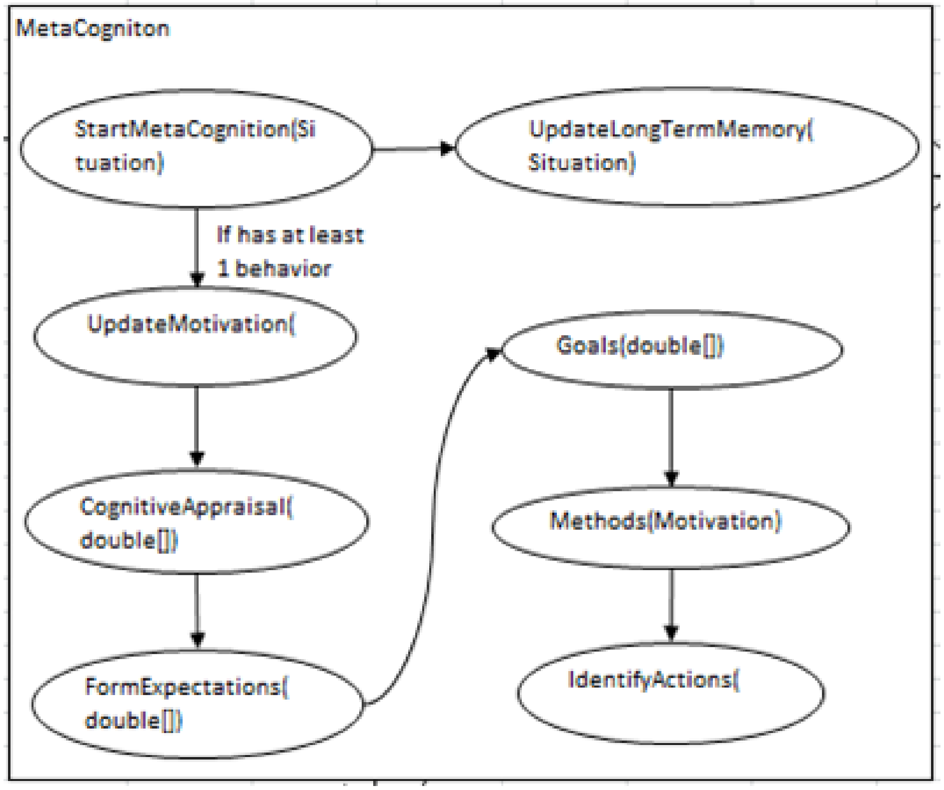

Figure C.10: Metacognition module

of the CognitiveArchitecture worksheet in the scenario file. If the agent has not performed that action the minimum threshold, ActionSelection choses EL for the agent. EL relies on another variable in Column M of the CognitiveArchitecture worksheet called temperature. If temperature is close to zero (0), an agent will be in an "exploit" mode - doing what the agent knows will result in the highest utility. If the temperature is set higher, agents will be in a "explore" mode. While in explore mode, agents may make decisions at random to "learn" about new options. The temperature variable decreases over time during a model run, thus enabling agents to "learn" to maximize their utility when making decisions.

Once an agent makes a decision via the SelectAction node, the model envokes that method to conduct the chosen action in the InvokeMethod node. This method could send a communication, seek a resource, or do nothing. Agents can communicate about events that happen to them in the model or about infrastructure status (success or failure). When these communications occur they are sent as Percept objects for other agents in the model to receive. Communications are based on social distance (homophilly), so agents will only communicate with other agents that share a sufficient link weight, determined by demographic dimensions. The communication link weight threshold is in the SimpleActionUmpire worksheet in the scenario file. After the agent conducts an action (or no action) the cognitive architecture has completed one iteration, the Agent Component (Figure C.13) of the cognitive architecture is listening for the source InvokeMethod, and the entire cognitive architecture process begins again.

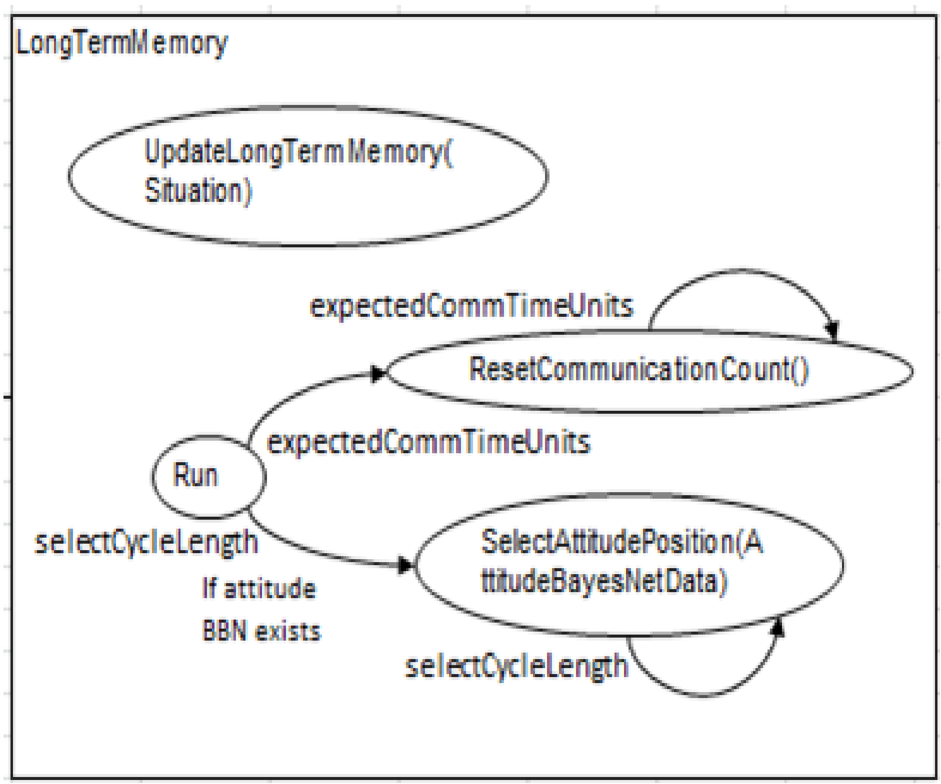

Figure C.11: Long Term Memory module

UID 3.2.1 Population model shall include formal documentation for data requirements used to generate all initialization conditions and other model parameters

The SIM 1.0 has a User Guide (Appendix H). Ms. Kristen Clark , the TRAC-WSMR SIM point of contact (POC), learned how to use the User Guide during her visiting analyst tour at TRAC-MTRY. TRAC-MTRY trained additional WSMR personnel during the week-long SIM Transition training event from 17-21 September 2012. Past technical reports on IW TWG also explain the process, and TRAC-MTRY delievered these past reports with the final delivery of SIM. Finally, Appendix J of this document contains an alternative method to design scenarios. The SIM Transition Technical Report (this document) includes both the User Guide and SIM 3.0 data-design methodologies. Transition included present and past documentation for:

- Data development proceedures.
- Population partition techniques.
- Subject Matter Expert (SME) elicitation proceedures.
- Bayesian belief network (BBN) development.
- Case file development.

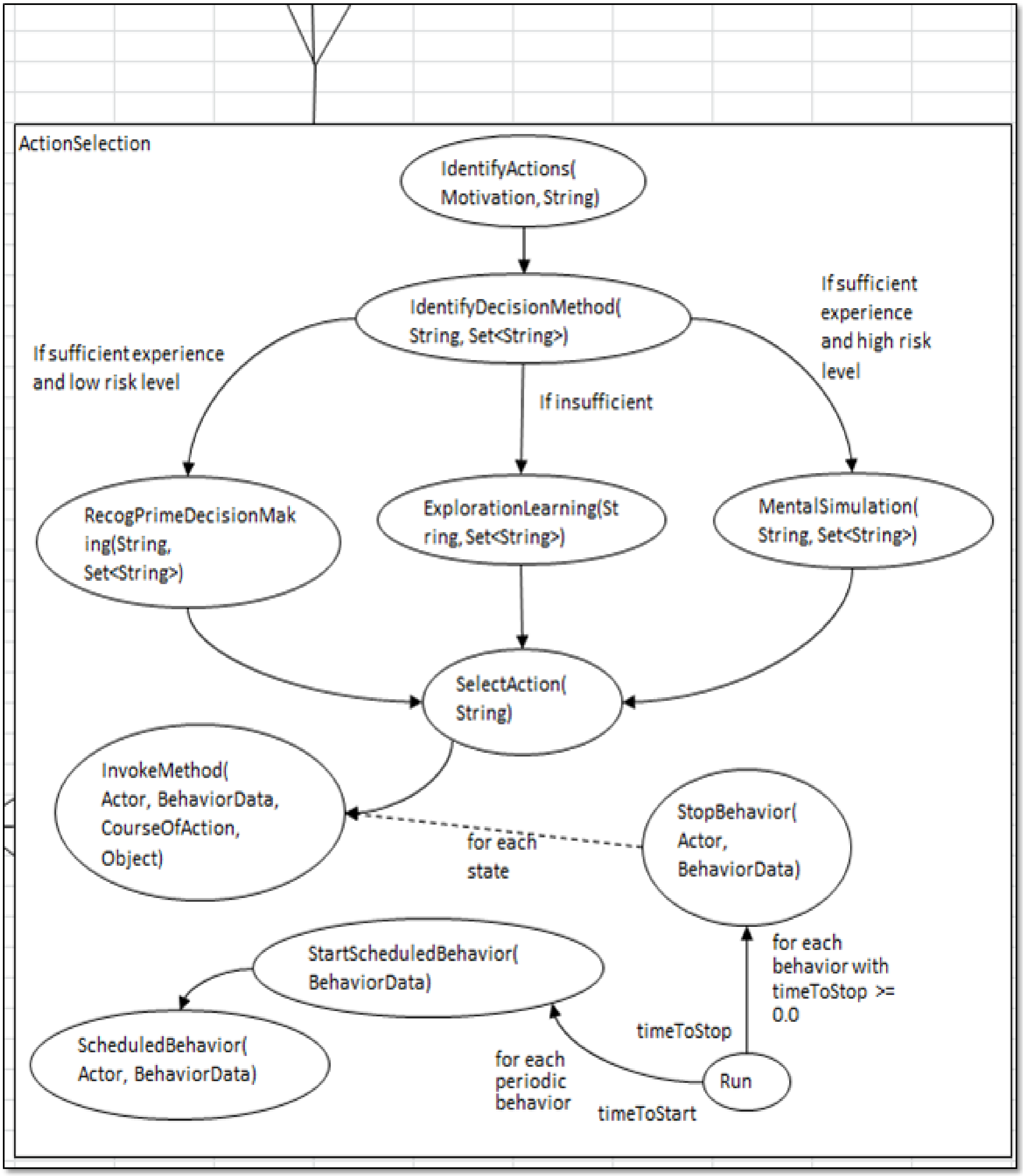

Figure C.12: Action Selection module

- Scenario file development.

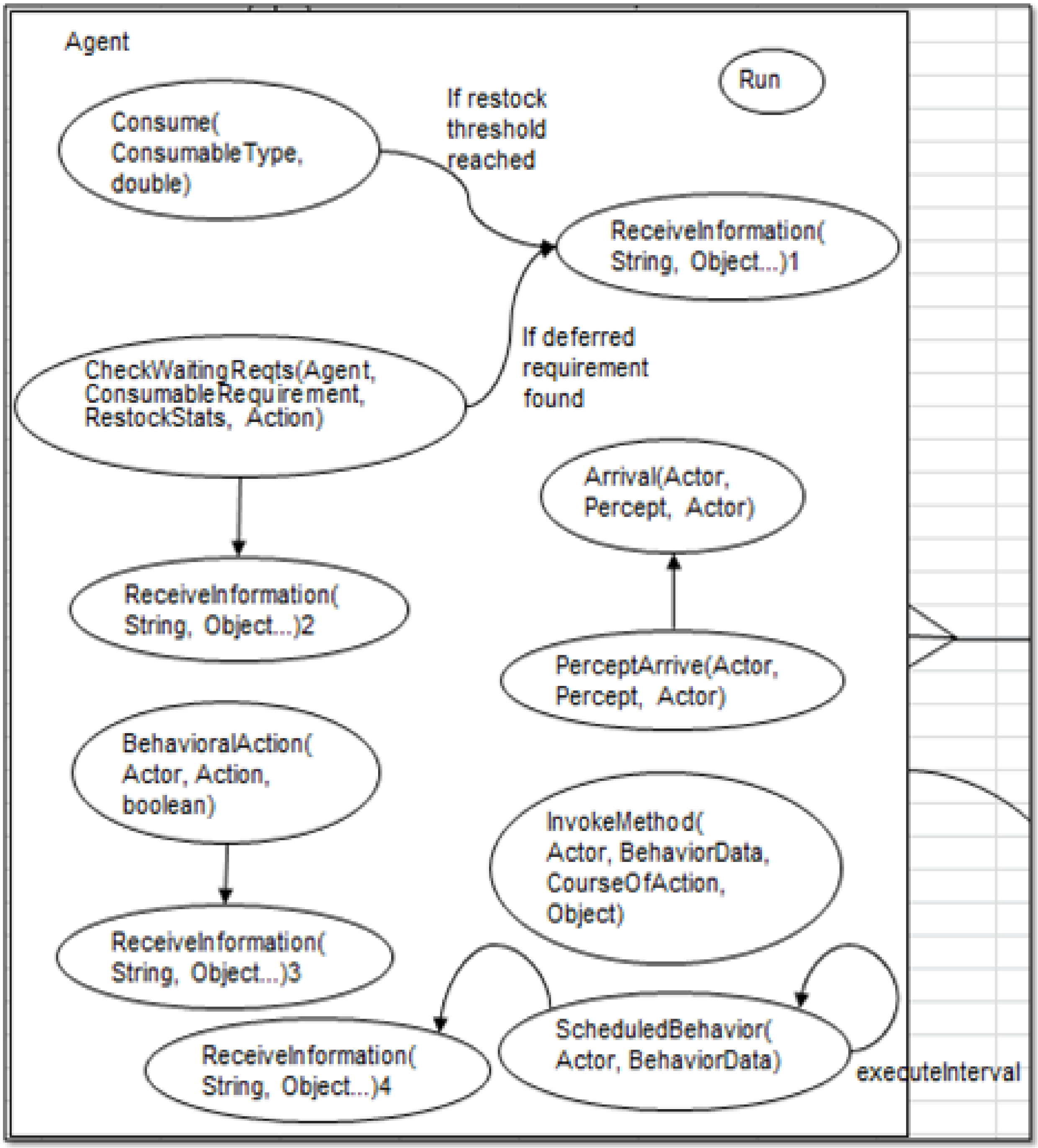

Figure C.13: Agent Component "listening" to ActionSelection Component

C.2. CONCLUSION

The cognitive architecture is the heart of SIM. The components mentioned in this appendix represent how agents perceive their environment, process information, and decide how they feel about the world. The development of SIM 2.0 and 3.0 required only minor modifications to the design described in this appendix. Future development of the population model in SIM focused on alternative modeling techniques for representing population issues and opinions. These developments were related to data and data development (scenario file) more than altering the code used to implement the scenario file.

APPENDIX D. SIM 1.0 TESTING RESULTS SAMPLING

D.1. OVERVIEW

This appendix explains the testing methodology utilized by the SIM Transition team. Testing began with an evaluation of SIM 1.0. SIM 1.0 testing focused exclusively on the population model because the contract for Nexus support expired. A plan was already in place to implement the key leader capability in SIM 2.0, so there was not a need to do further testing of Nexus code. Results from the IW TWG 2011 served as a baseline for future testing. This appendix provides a look at the most significant results from testing the model, and answers several specific questions asked in the requirements document (Appendix A).

D.2. TEST AND EVALUATION METHODOLOGY

The SIM Transition team initially employed a top-down approach to testing. After several iterations, the team realized that this method of testing was not going to produce results that could be easily verified against expected outputs from the model due to the complexity of multiple agents, events, and infrastructure. The team switched to a bottom-up approach whereby the simplest of scenarios were tested and additional complexity was added to verify results. The bottom-up approach allowed the team to declare the expected results, test the model, and trace varibales throughout the code to ensure SIM performed to standard.

D.3. RESULTS

D.3.1. Specified Tests in the requirements document

CG UID 1.2.0 Provide the results of a design of experiment (DOE) on the scenario events (SE) in order to determine impacts that different quantities have on the population.

The development team designed an extensive DOE for testing the impacts of scenario events on the population. This testing focused on various quantities of scenario events and agents to demonstrate the effects on population opinions and beliefs. Table D.1 below is a simple example DOE used for early iterations of SIM 1.0 testing. This technique was reused for all phases of testing after SIM 1.0. Table D.2 shows a much larger DOE used to establish a baseline for future testing. TRAC-WSMR received a complete DVD of test cases on 21 September 2012, including over 1600 input scenario files and the resulting output, in excess of 17,000 files. Of these files, over 200 input and over 2600 output files were from SIM 1.0 testing.

Agents	Stereotype(s)	Location	Scenario Events	Infrastructure	Test
1	1	1 Hex	1 (e.g. Patrols)	None	OAB / Issue Stance
2	1	1 Hex	1 (e.g. Patrols)	None	OAB / Issue Stance
2	2 Similar	1 Hex	1 (e.g. Patrols)	None	OAB / Issue Stance
2	2 Dissimilar	1 Hex	1 (e.g. Patrols)	None	OAB / Issue Stance
1	1	1 Hex	10 (e.g. Patrols)	None	OAB / Issue Stance
2	1	1 Hex	10 (e.g. Patrols)	None	OAB / Issue Stance
2	2 Similar	1 Hex	10 (e.g. Patrols)	None	OAB / Issue Stance
2	2 Dissimilar	1 Hex	10 (e.g. Patrols)	None	OAB / Issue Stance

Table D.1: Basic Experimental Design for Testing Scenario Events.

After MAJ Brown completed training LTC Caldwell how to use the scenario file during the first iteration of testing, the team developed an initial DOE to guide the second iteration of SIM 1.0 testing. Table D.1 shows the basic methodology employed. As discussed in Chapter 2 of this report, this initial strategy sought to peal away complexity in the model and isolate resulting output. Results were traceable; however, due to case files that lacked evidence from the survey data, the results were still difficult to interpret (Figure D.1).

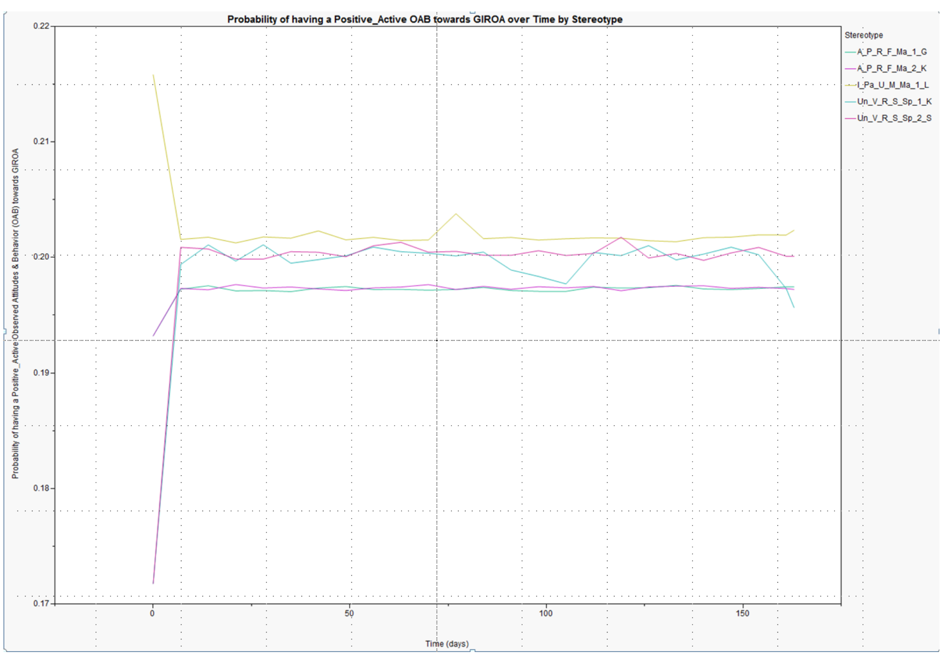

Figure D.1: SIM 1.0, Iteration 2 Testing Results

After manipulating TWG 2011 scenario files and having marginal results, the team decided that a bottom-up approach was more appropriate. The necessitated a complete restart, and developers began building a set of scenario files from the ground up. Table D.2 outlines the target variables in the first set of scenario files. Those variables were:

- Discount factor (λ).
- Number of survey respondents in the case file.
- Mix of beliefs and interests.
- Observed attitudes and behaviors.
- Positive or negative events (one per day).

Design Point	Survey	Belief		Issue Stance		OAB					Discount	Scripted Action	Infrastructure	SIM Time	Effect Direction
	# Resp.	Adequate	Inadequate	Adequate	Inadequate	NA	NP	N	PP	PA	λ				
1	100	1	99	1	99	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Positive
2	100	99	1	99	1	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Positive
3	100	50	50	50	50	0.5000				0.5000	0.9900	CFOperatesInArea	N/A	140	Positive
4	100	50	50	50	50	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Positive
5	100	50	50	50	50	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Positive
6	100	100	0	100	0	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Positive
7	100	0	100	0	100	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Positive
8	100	1	99	1	99	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Negative
9	100	99	1	99	1	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Negative
10	100	50	50	50	50	0.5000				0.5000	0.9900	CFOperatesInArea	N/A	140	Negative
11	100	50	50	50	50	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Negative
12	100	50	50	50	50	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Negative
13	100	100	0	100	0	0.0100				0.9900	0.9900	CFOperatesInArea	N/A	140	Negative
14	100	0	100	0	100	0.9900				0.0100	0.9900	CFOperatesInArea	N/A	140	Negative
15	100	1	99	1	99	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Positive
16	100	99	1	99	1	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Positive
17	100	50	50	50	50	0.5000				0.5000	0.5000	CFOperatesInArea	N/A	140	Positive
18	100	50	50	50	50	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Positive
19	100	50	50	50	50	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Positive
20	100	100	0	100	0	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Positive
21	100	0	100	0	100	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Positive
22	100	1	99	1	99	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Negative
23	100	99	1	99	1	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Negative
24	100	50	50	50	50	0.5000				0.5000	0.5000	CFOperatesInArea	N/A	140	Negative
25	100	50	50	50	50	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Negative
26	100	50	50	50	50	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Negative
27	100	100	0	100	0	0.0100				0.9900	0.5000	CFOperatesInArea	N/A	140	Negative
28	100	0	100	0	100	0.9900				0.0100	0.5000	CFOperatesInArea	N/A	140	Negative
29	100	1	99	1	99	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Positive
30	100	99	1	99	1	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Positive
31	100	50	50	50	50	0.5000				0.5000	0.1000	CFOperatesInArea	N/A	140	Positive
32	100	50	50	50	50	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Positive
33	100	50	50	50	50	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Positive
34	100	100	0	100	0	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Positive
35	100	0	100	0	100	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Positive
36	100	1	99	1	99	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Negative
37	100	99	1	99	1	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Negative
38	100	50	50	50	50	0.5000				0.5000	0.1000	CFOperatesInArea	N/A	140	Negative
39	100	50	50	50	50	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Negative
40	100	50	50	50	50	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Negative
41	100	100	0	100	0	0.0100				0.9900	0.1000	CFOperatesInArea	N/A	140	Negative
42	100	0	100	0	100	0.9900				0.0100	0.1000	CFOperatesInArea	N/A	140	Negative
43	100	1	99	1	99	0.9900				0.0100	0.0100	CFOperatesInArea	N/A	140	Positive
44	100	99	1	99	1	0.0100				0.9900	0.0100	CFOperatesInArea	N/A	140	Positive
45	100	50	50	50	50	0.5000				0.5000	0.0100	CFOperatesInArea	N/A	140	Positive
46	100	50	50	50	50	0.9900				0.0100	0.0100	CFOperatesInArea	N/A	140	Positive
47	100	50	50	50	50	0.0100				0.9900	0.0100	CFOperatesInArea	N/A	140	Positive
48	100	100	0	100	0	0.0100				0.9900	0.0100	CFOperatesInArea	N/A	140	Positive
49	100	0	100	0	100	0.9900				0.0100	0.0100	CFOperatesInArea	N/A	140	Positive
50	100	1	99	1	99	0.9900				0.0100	0.0010	CFOperatesInArea	N/A	140	Negative
51	100	99	1	99	1	0.0100				0.9900	0.0010	CFOperatesInArea	N/A	140	Negative
52	100	50	50	50	50	0.5000				0.5000	0.0010	CFOperatesInArea	N/A	140	Negative
53	100	50	50	50	50	0.9900				0.0100	0.0010	CFOperatesInArea	N/A	140	Negative
54	100	50	50	50	50	0.0100				0.9900	0.0010	CFOperatesInArea	N/A	140	Negative
55	100	100	0	100	0	0.0100				0.9900	0.0010	CFOperatesInArea	N/A	140	Negative
56	100	0	100	0	100	0.9900				0.0100	0.0010	CFOperatesInArea	N/A	140	Negative

Table D.2: Baseline Testing: First 56 Design Points

Figure D.2 shows the results of the fifth iteration of SIM 1.0 testing that established a baseline for all future testing. This was the first result where the team saw the dramatic

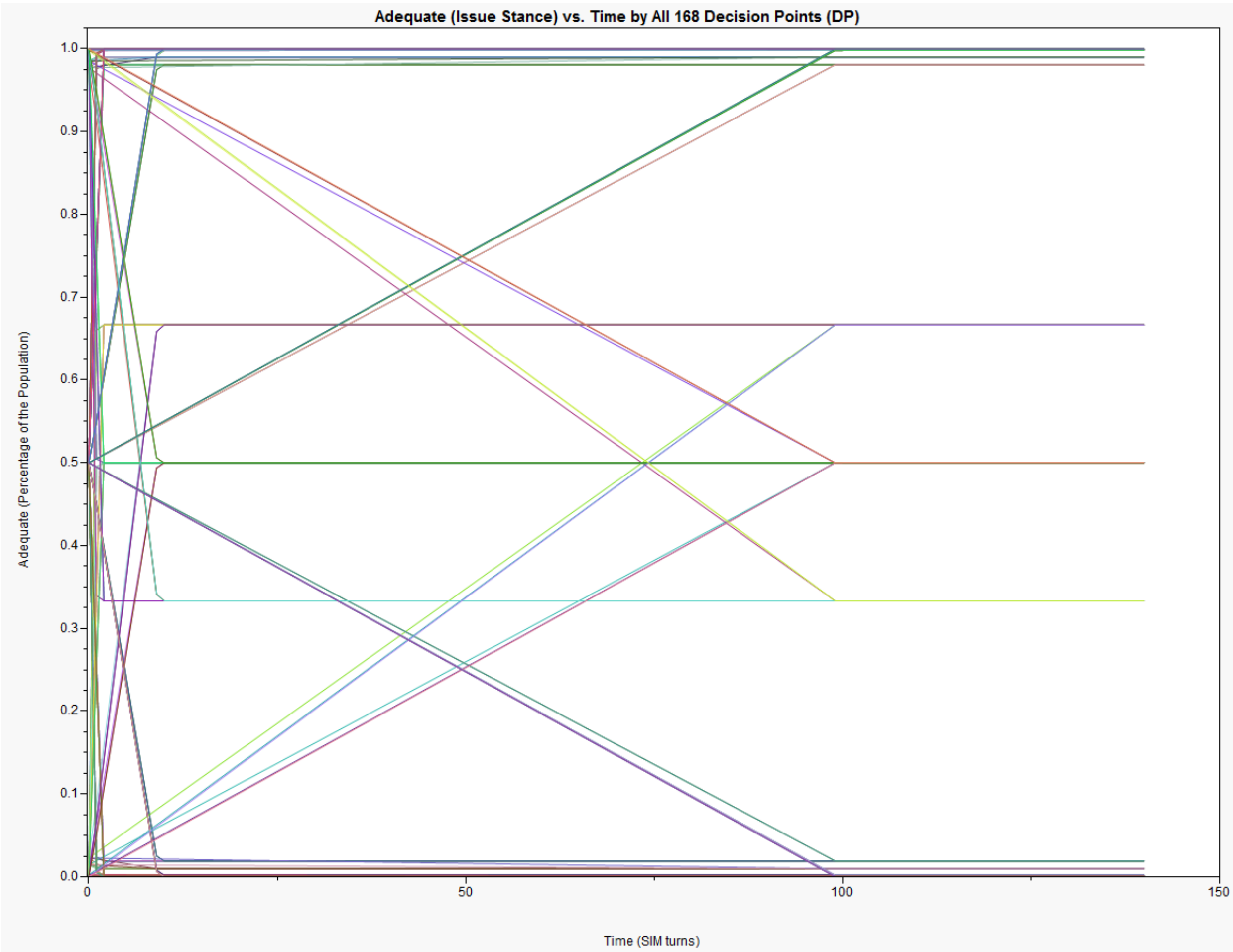

Figure D.2: Baseline Testing: SIM 1.0, Iteration 4

impact of the discount factor on model results. In Figure D.2, the lines that converge near time-step 100 are when the discount factor was 0.01 - the recommended setting. The different lines converging at those points (% of the population agents who believed that the Civil Security issue stance was adequate) varied by the initial condition of the population. In every instance there are three lines that are very close together. Those lines were the survey respondent variables (100, 1000, and 10,000 for this iteration). They did not have a large impact, and the team investigated those further breifly with SIM 1.0 and in more depth with SIM 2.0 testing. The remainder of this appendix focuses on SIM 1.0 testing that demonstrated the effects of the discount factor (λ) and a population stereotype's initial issue stance.

In order to control variables in the baseline tests, the design team created a set of case files containing specific proportions of survey respondent's answers. The team built BBN where one belief (Security) informed one issue (Civil Security). The names of the belief and issue are not important, they could easily be called "x" and "y". The case files were developed on the extremes in order to demonstrate the minimum or maximum limits for model output, given the controlled conditions. Table D.2 shows the breakdown of beliefs and issues for the various case files. For example, DP_1 utilized a case file for a population stereotype containing 100 respondents. In that case file, there is one respondent who believed that security was adequate, and that same respondent believed that civil security was also adequate. The other 99 respondents felt neither was adequate. This combination created an extreme case for testing the effect of an event mapped to a 100% positive impact on security. Figure D.3 shows the resulting outcome.

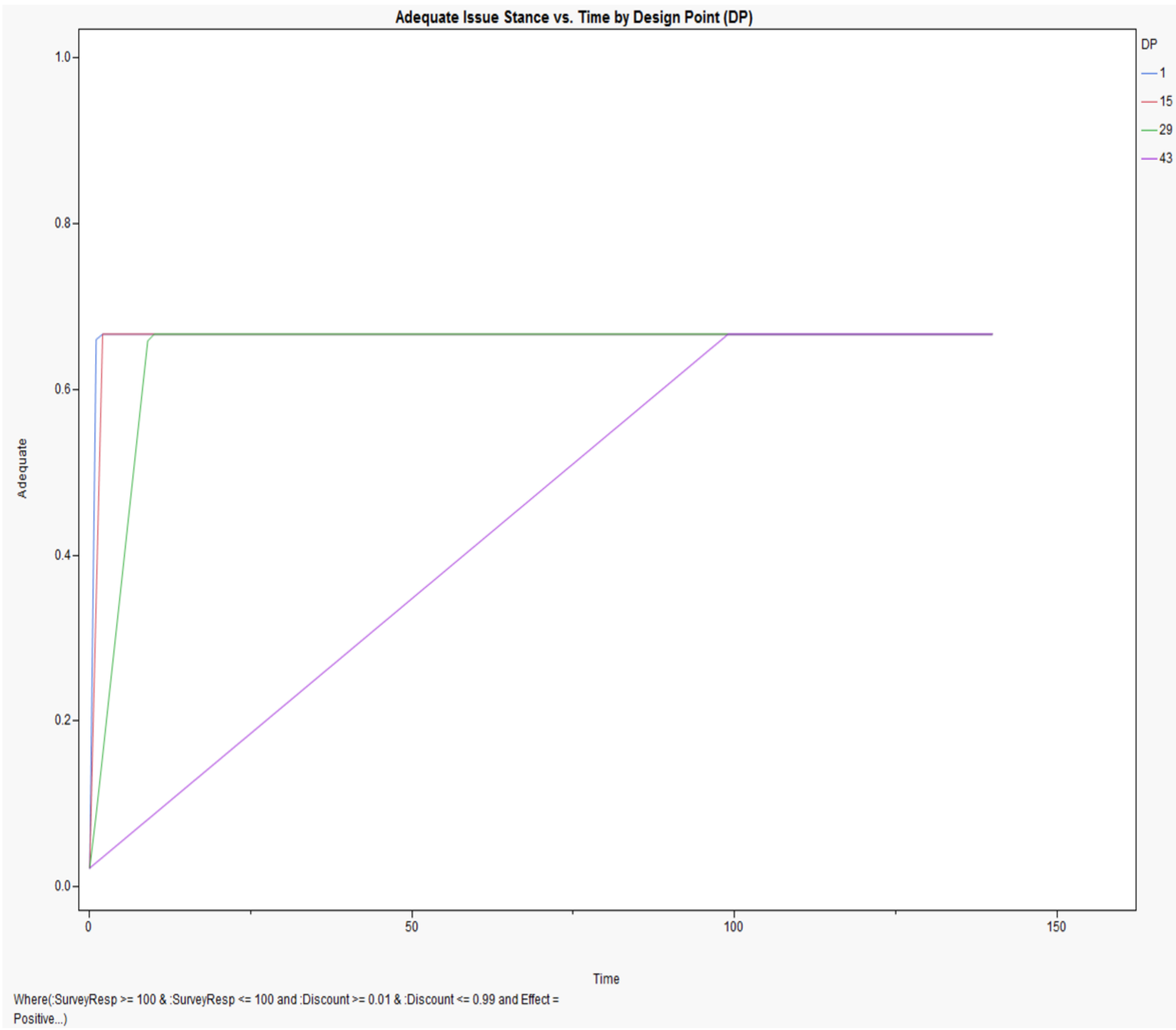

Figure D.3: Varying λ: Positive Events and 1% Adequate Population Issue Stance

Figure D.3 shows that there is a huge difference between a λ of 0.01 (DP_43) and 0.99, 0.5, or 0.1 (DP_1, DP_15, and DP_29). It also demonstrates the asymptotic behavior of the BBN. There is always evidence in the BBN of initial issue stances, so even with a discount factor, there is a maximum opinion output from the model. Under the conditions for these DP, the maximum population opinion of civil security is 67% adequate.

Figure D.4 show the same conditions; however, the initial case files flip from a very negative opinion of civil security to an extremely positive one - 99% adequate.

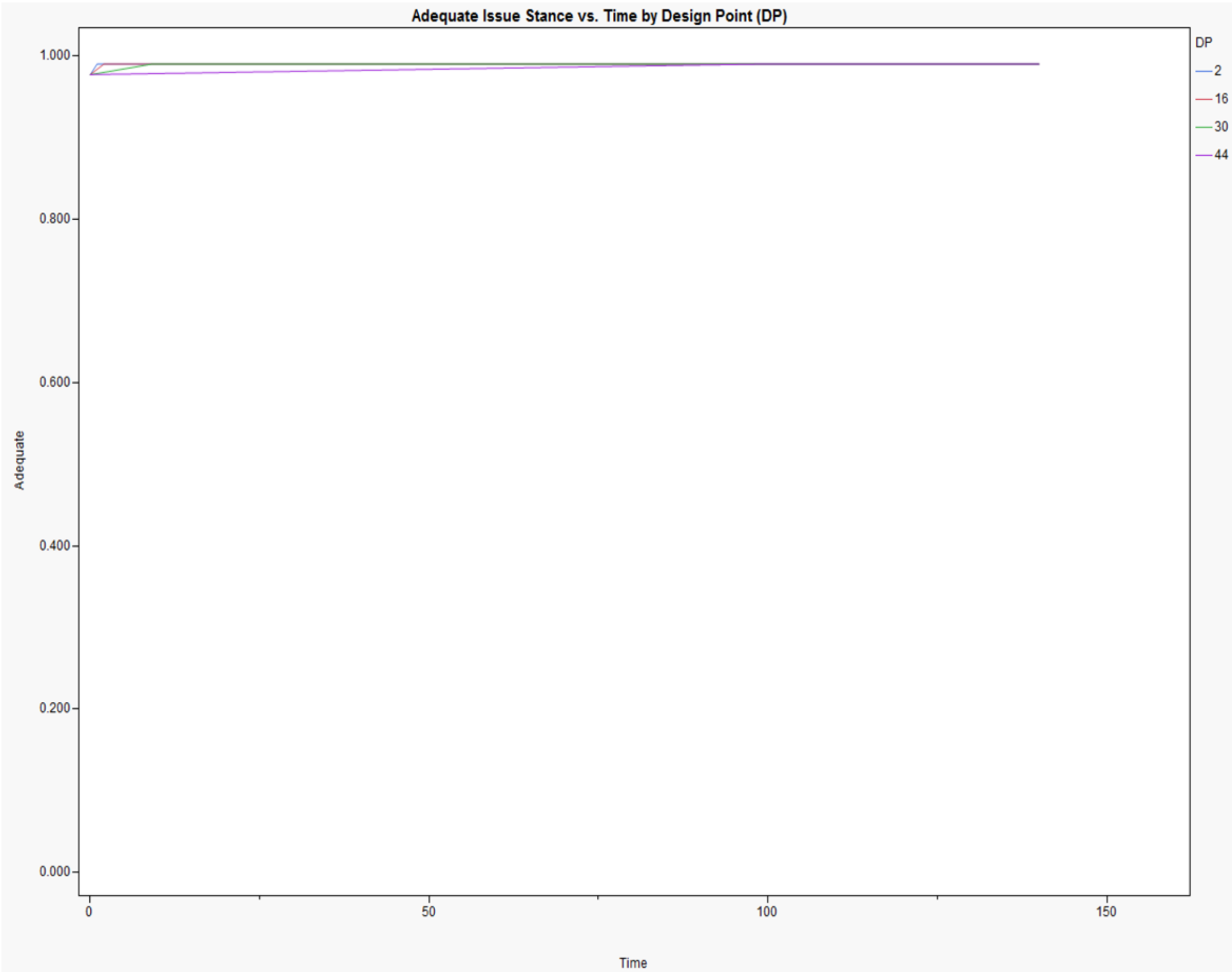

Figure D.4: Varying λ: Positive Events and 99% Adequate Population Issue Stance

Figure D.4 demonstrates that the population issue stance will approach 100% adequate, given a population stereotype with a very high opinion of the issue stance in question. This result is intuitive; however, the team believed that it was important to show that the model behaved as expected, given very little room to improve an issue stance.

The next three tests looked at 100 population stereotype respondents evenly split on an issue. The case files contained 50% of the population that believed security was adequate and the issue stance (Civil Security) was 50% adequate. The remaining 50 repondents had a belief and issue stance of inadequate. The three cases only varied by the underlying OABs - two on the extremes and one in the middle. Those vairations were:

- 1% Positive Active and 99% Negative Active
- 99% Positive Active and 1% Negative Active

- 50% Positive Active and 50% Negative Active

Figure D.5, D.6, and D.7 demonstrate that the population issue stance will approach 100% adequate, given a population stereotype with a 50-50 initial issue stance. This is encouraging, but these three tests also reveal that the OAB do not affect issue stances. SIM is well suited to model population issue stances; however, modeling OAB should require a different process or data that asks specific questions about attitudes toward actors. The research team believes that population issue stances should influence OAB. In SIM 1.0, this is not the case, and previous TWG use cases have employed SIM 1.0.

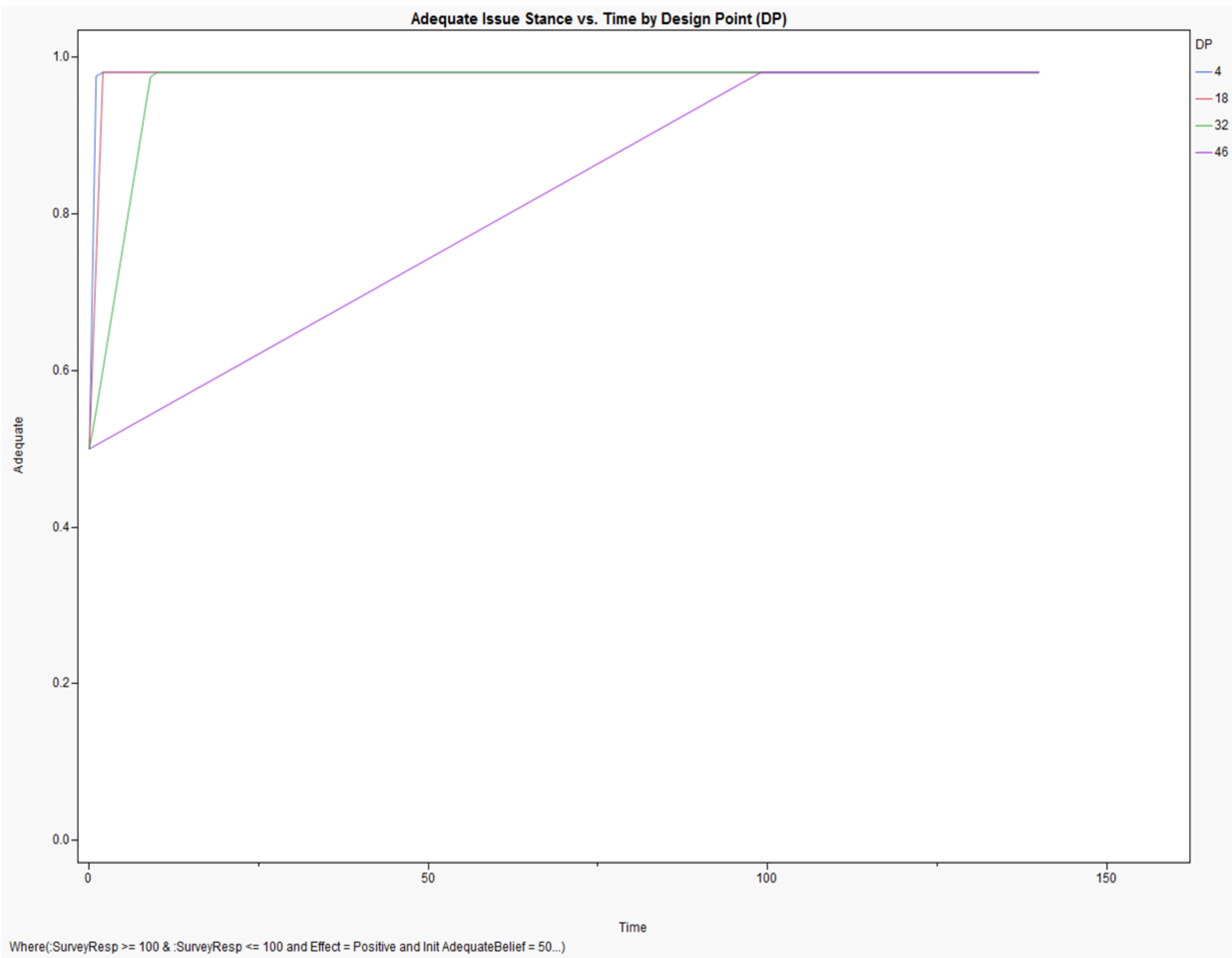

Figure D.5: Varying λ: Positive Events, 50% Adequate Issue Stance, and 99% NA OAB

After testing the even mix of issue stances on civil security, the team wanted to evaluate “absolue” evidence. TRAC-MTRY built case files containing no evidence of adequacy and files with no inadequate beliefs. Using the same construct as the previous tests, Figure D.8 demonstrates that the population won’t move off 100% adequate, if they begin in that state. It is important to note here that if the effect of the positive event would have been a fraction lower, the issue stance would start to decrease. Appendix F and Appendix G will discuss the reason for this phenomenon.

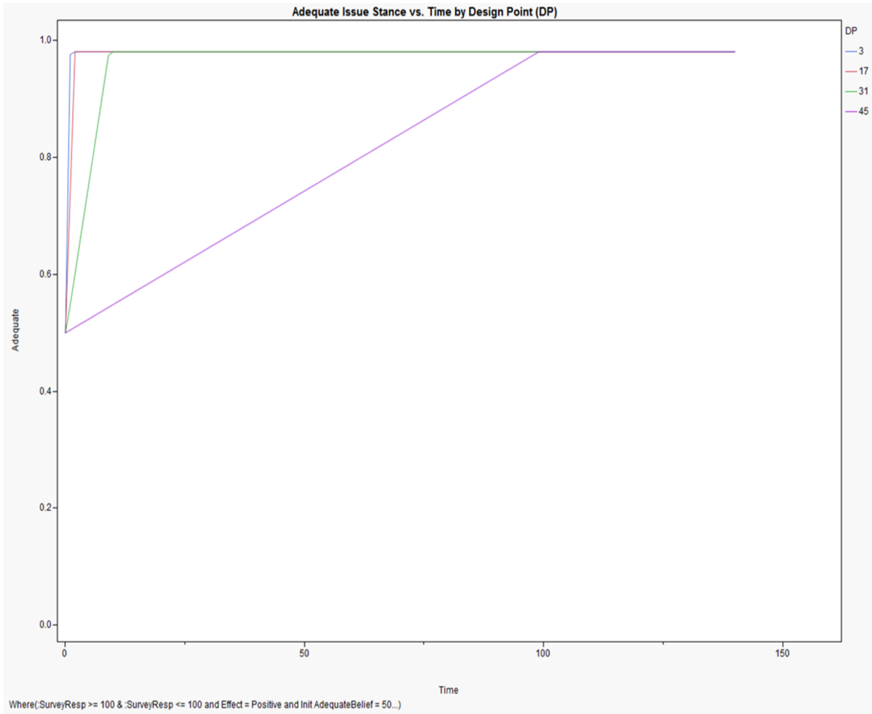

Figure D.6: Varying λ: Positive Events, 50% Adequate Issue Stance, and Split OAB

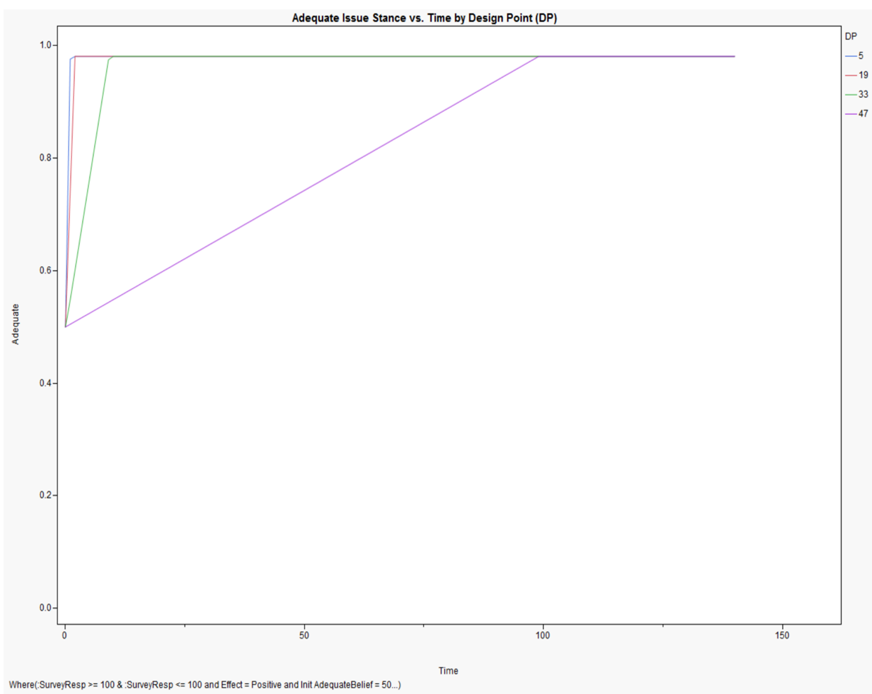

Figure D.7: Varying λ: Positive Events, 50% Adequate Issue Stance, and 99% PA OAB

In a similar manner, the team tested a population that showed no evidence of feeling that the issues were in an adequate state. Figure D.9 shows that a population on the furthest

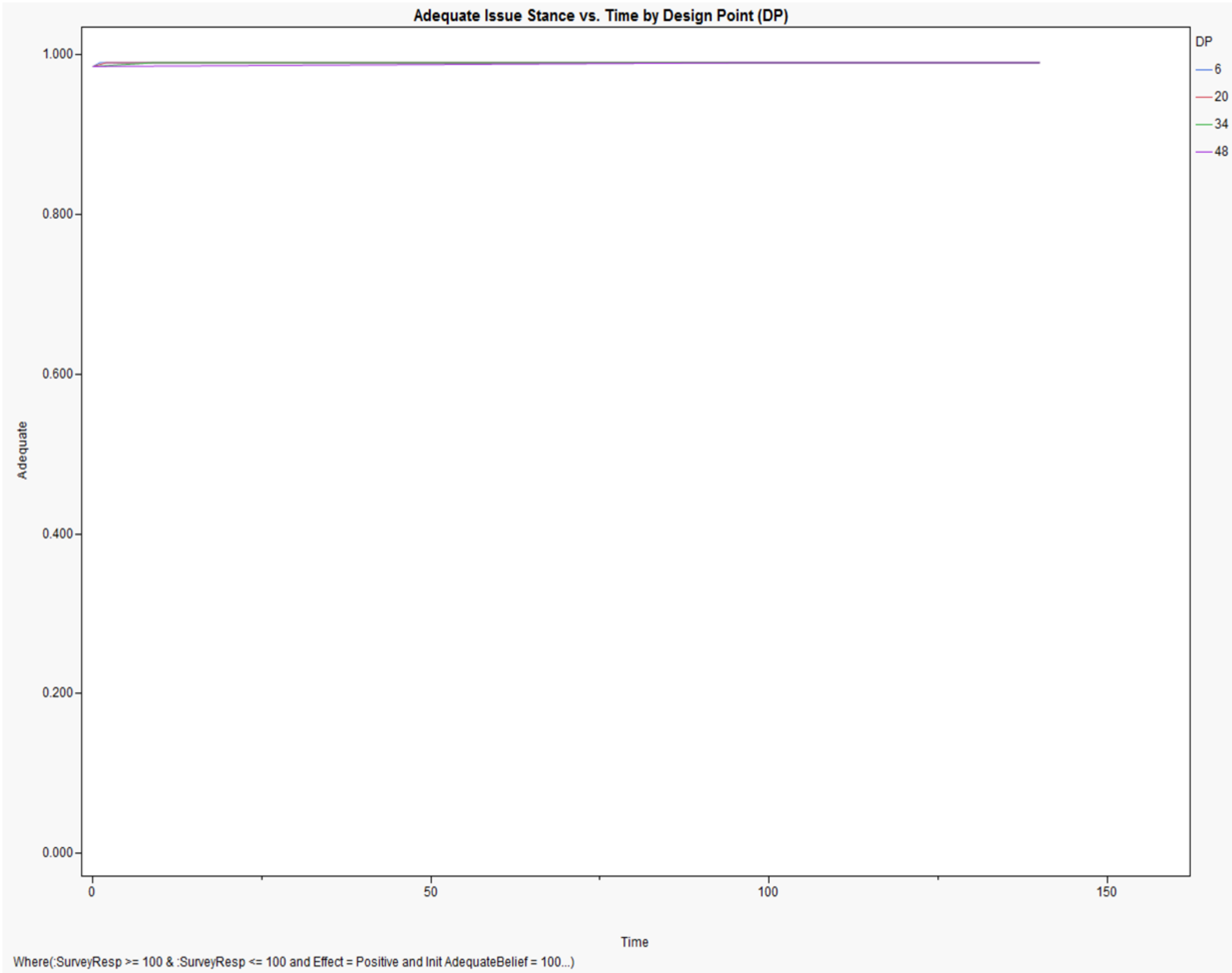

Figure D.8: Varying λ: Positive Events and 100% Adequate Population Issue Stance

extreme can be turned over time to reach a 50% adequate issue stance with events having a 100% positive effect on the population.

D.3.2. Negative Effects Testing

Finally, the team verified that negative actions had an opposite reaction on the population. All conditions remained the same for the next set of tests; however, the team mapped a 100% negative effect to the event. The same result would come from mapping a 0% positive effect of any given event; however, the team believed that 100% negative made more sense and was a stronger statement when compared to 0% positive. Figure D.10 shows the effect of negative effects on a population that has an initial issue stance of 99% adequate.

In the same manner, the team looked at 100% negative effects on a population that already had a low opinion of the civil security issue stance (1% adequate). Much like the positive case shown in Figure D.4, Figure D.11 has little room to decline, but the issue stance will approach 0% adequate because the effect is still lower than the 1% adequate initial issue

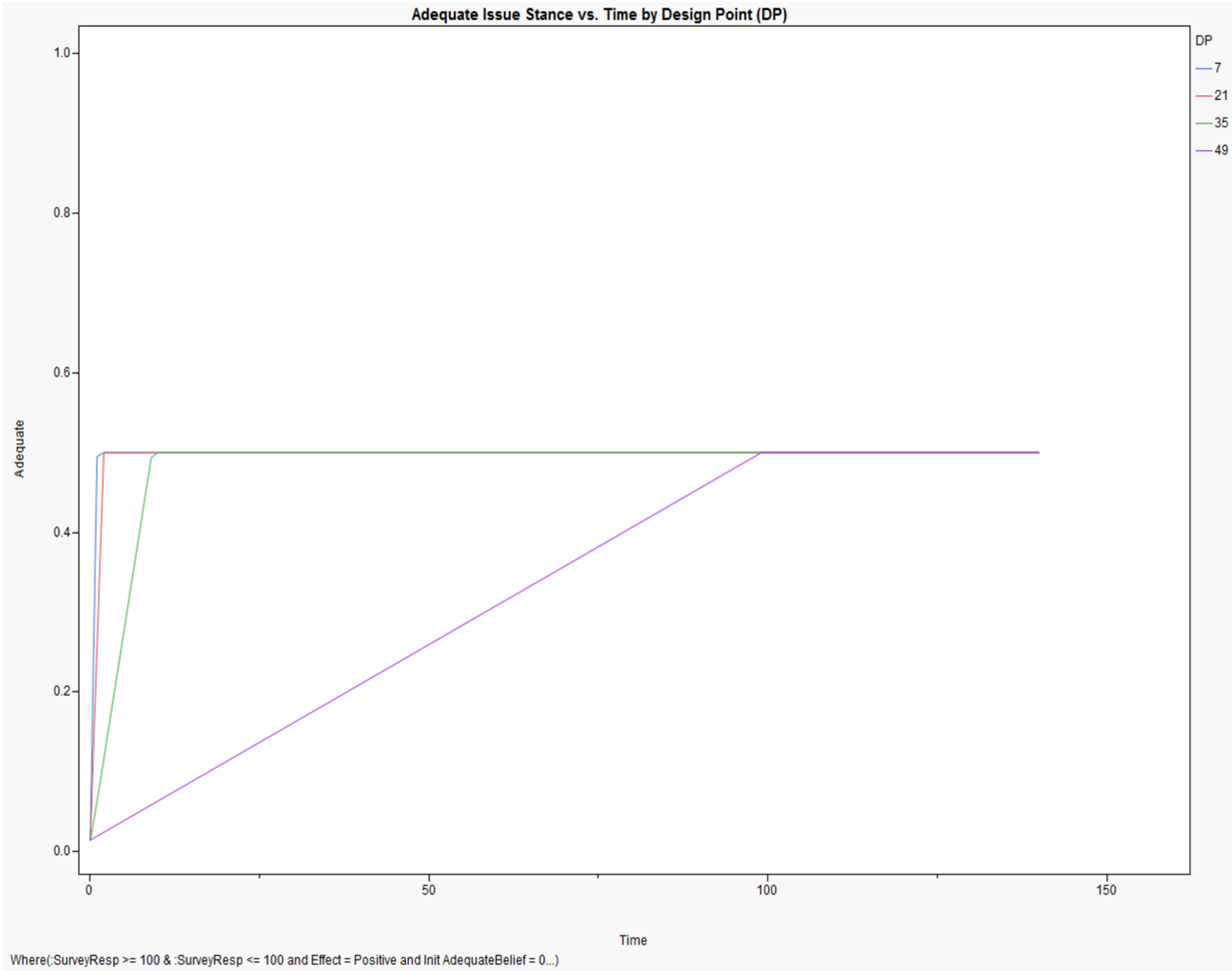

Figure D.9: Varying λ: Positive Events and 0% Adequate Population Issue Stance

stance.

UID 4.2.0 The population model shall provide population feedback that is consistent with the common operating picture uniquely seen by units operating in interactive complex environments (As described in FM 3.0 and FM 5.0)

The testing with SIM 1.0 verified that the if the IW TWG team choses to modify the effects of events, they can tailor results from the model to stimulate wargame players by calibrating the models. Therefore, the development team strongly recommends a calibration exercise to ensure that results meet the need for the next IW TWG.

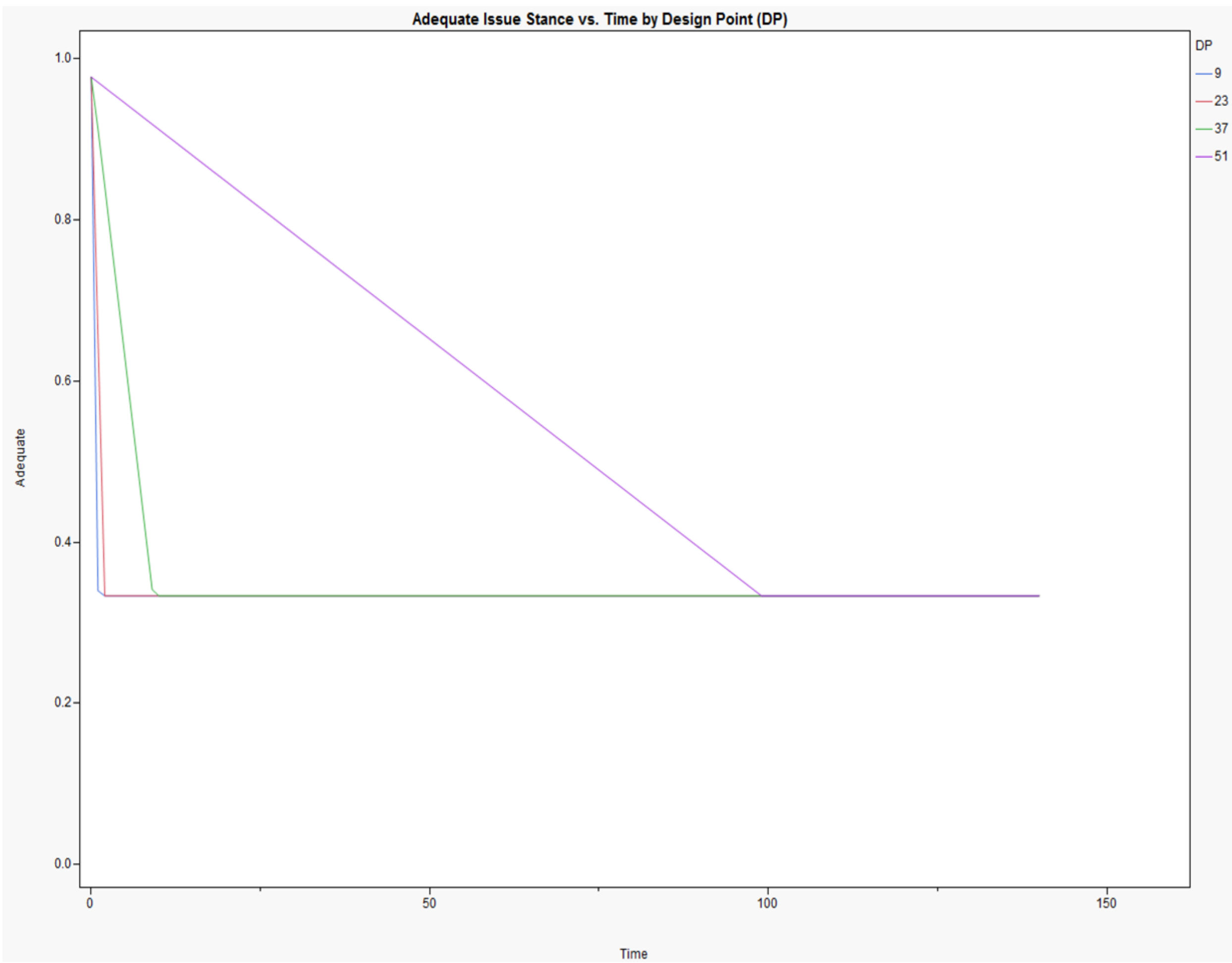

Figure D.10: Varying λ: Negative Events and 99% Adequate Population Issue Stance

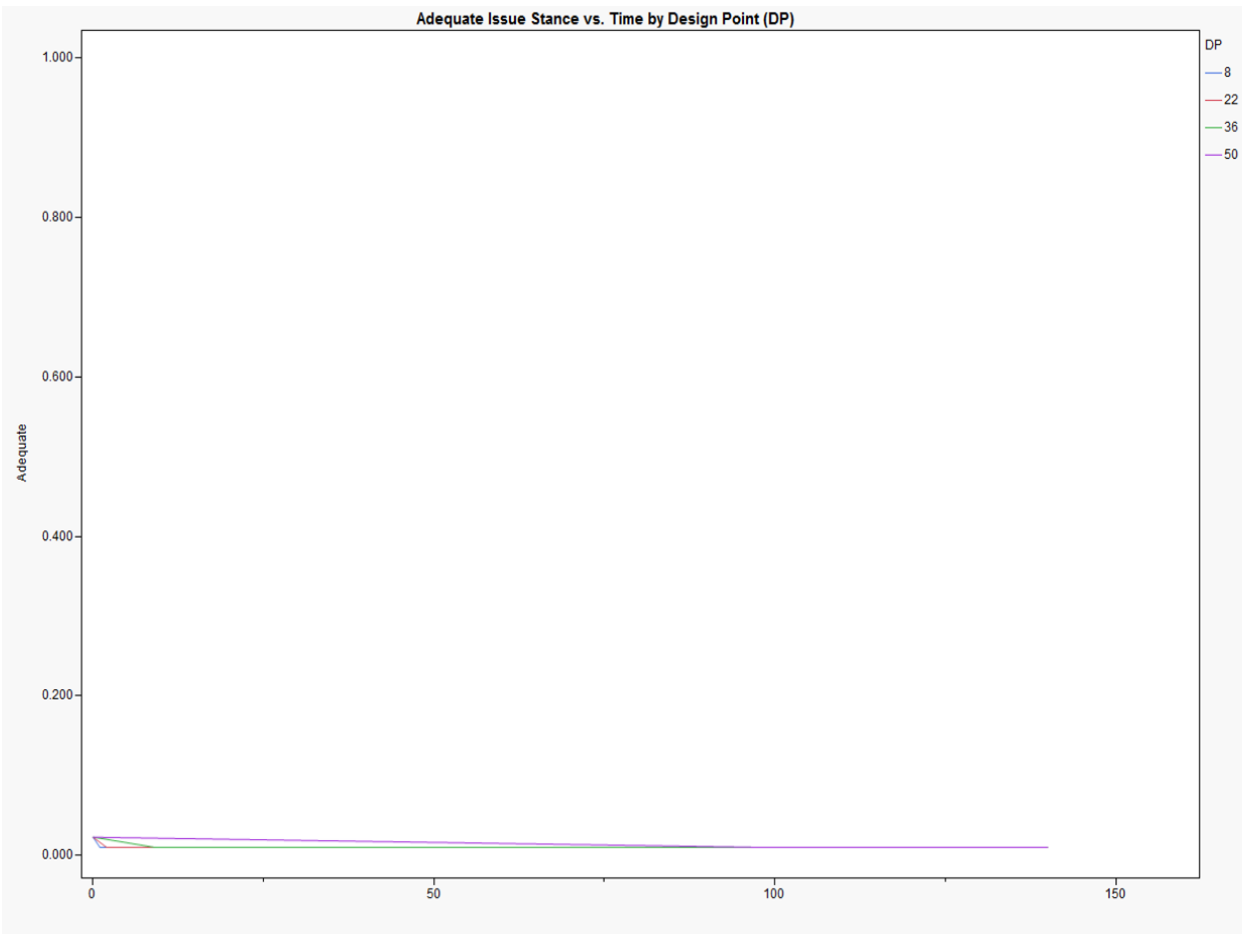

Figure D.11: Varying λ: Negative Events and 1% Adequate Population Issue Stance

D.4. CONCLUSIONS

SIM 1.0 testing revealed a good methodology for testing SIM in future iterations. Testing revealed that testing the extreme values uncovered the minimum and maximum results from the model. By controlling the CPT in the BBN, the team could verify the results against expected values. All tests passed and TRAC-MTRY carried the 168-baseline tests forward as the first iteration of testing for SIM 2.0.

APPENDIX E. SIM 2.0 TECHNICAL DESIGN

The primary design difference in SIM 2.0 is the addition of the key leader and social networks capability. In SIM 1.0 this capability was in the Nexus Network Learner model. In SIM 2.0, the same model that produces population issue stances and opinions also contains additional functionality:

- Establishes static networks (threat, friendly or neutral).
- Generates dynamic networks to perform behaviors specified in the scenario file.
- Conducts key leader engagements between TWG players and key leaders in the static networks in response to instructions from PAVE.
- Allows players to interview entities in the dynamic network.
- Contains the ability to conduct an engagement with multiple key leaders (Shura).
- Removes key leaders from networks based on PAVE instructions.
- Replaces key leaders in the static network based on an algorithm that uses a potential replacement's qualifications (roles) and affinity to others in the network.

E.1. SPECIFIED REQUIREMENTS

TRAC-WSMR specified the following requirements for SIM. This section does not address all of the requirements, but rather, the requirements met specifically by SIM 2.0. See Appendix A for a complete list of requirements for SIM Transition. This section covers the design of the key leader and social network models within SIM 2.0. The event graphs show a complete blueprint for these components and this appendix provides an overview those components. See Appendix C for the technical design of the population model. Continually referring to the key leader and social network module is a mouthful, and the development team began calling the it the key leader engagement (KLE) module, or KLE for short. For the purposes of this appendix, the use of the acronym KLE also means the entire key leader and social network module in SIM 2.0.

UID 13.2.0 The social network model shall have the same functionality that existed for the IW TWG 2011 game

Developers of the SIM 2.0 key leader and social network components refactored SIM 1.0 Nexus code into SimKit-based code. The team was careful to ensure that all capability made it from SIM 1.0 to SIM 2.0. This was a lengthly process that resulted in additional capability in SIM 2.0 upon completion. The entry point to the new modules is via the CgEventsToFire table in PAVE. Previously PAVE had a Nexus_Instruction table; however, the TRAC-MTRY team requested that a single table contain all model instructions for the latest versions of SIM. Activity from the KLE modules still goes to an output table in PAVE called Nexus_Instruction_Result. Upon completion of SIM 2.0 this had not changed; however, in the future, it may be more appropriate to call it something other than Nexus.

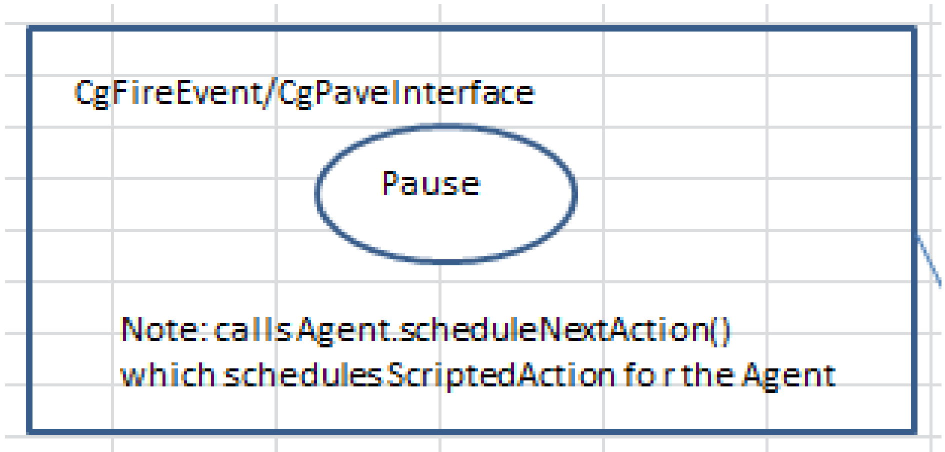

Figure E.1: Component Entry Point to Key Leader and Social Network Module (KLE)

Figure E.1 is a component listening for an event from the PAVE database. When a KLE action arrives, the scheduleNextAction method schedules a ScriptedAction event for the agent (key leader or entity) affected by the action. Figure E.2 shows the ScriptedAction node in the Agent Component. The arrow back to itself indicates that it is a self scheduling node, meaning that there may be events that schedule additional ScriptedAction events at another time (or simultaneously). The ScriptedAction node schedules the NextAction event. The Agent Component also contains the Killed event that would remove a key leader from the TWG; however, this will be addressed later in this appendix.

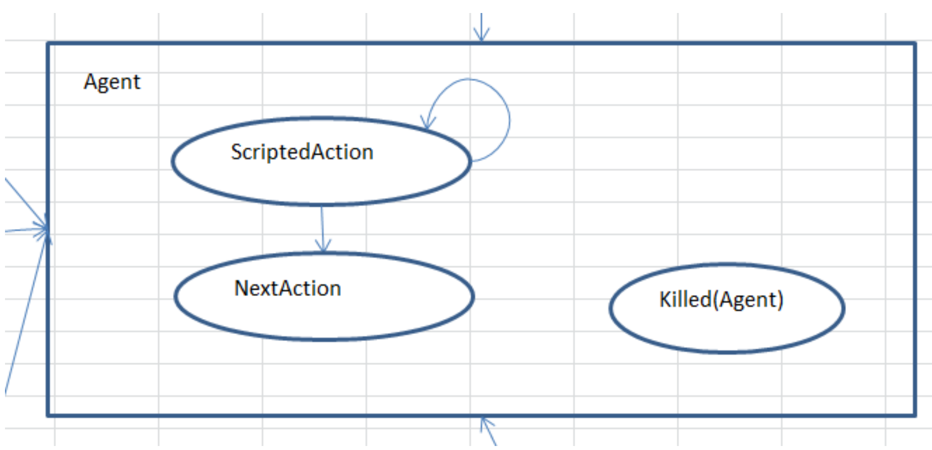

Figure E.2: Key Leader Agent Component

The ActionUmpire Component listens for a NextAction event. The use of Umpires in SimKit code ensure that events are handled and directed properly. The ActionUmpire begins a StartKeyLeaderEngagement Event or ReceiveInformation Event based on the action type. Separate umpires handle both these events Figure E.3 visually depicts this structure.

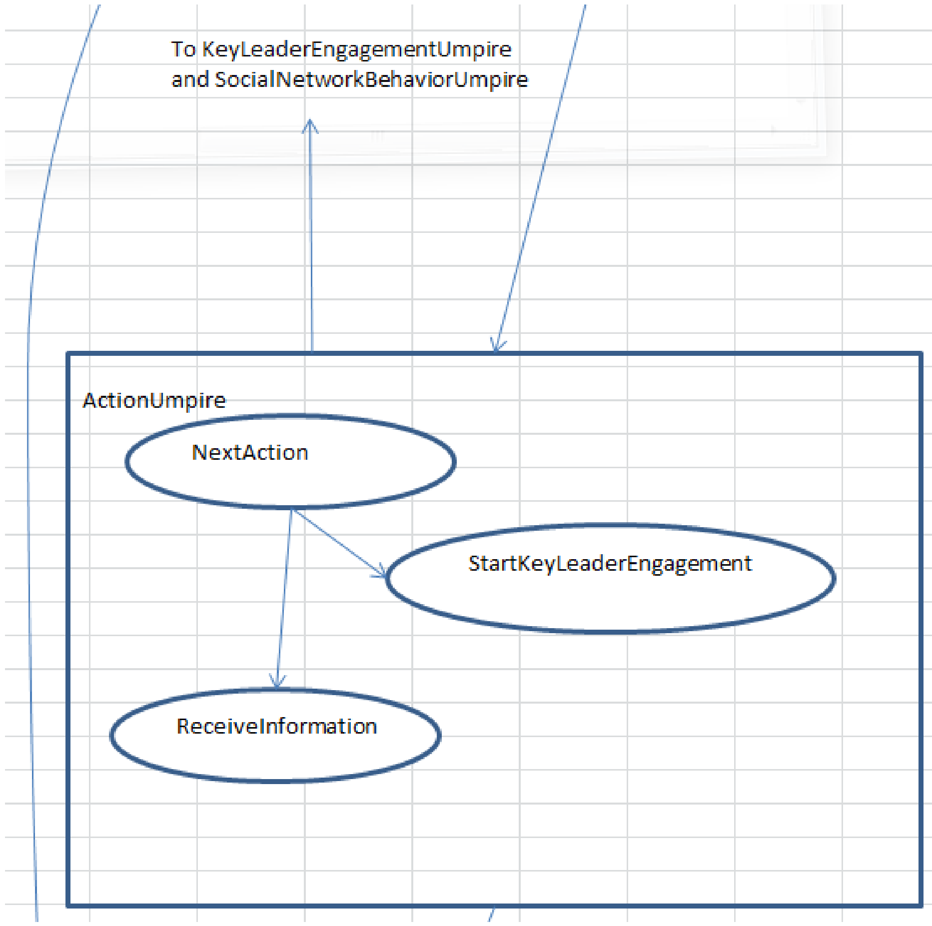

Figure E.3: Action Umpire Component

Figire E.4 shows the three types of reported events back to the PAVE database. A killed event reports key leaders removed from the TWG. A KLEInfo event updates results of information (critical knowledge) for interviews or other means. Finally, the ReplacedVacancy event reports changes in the static networks. These three nodes listen for events to fire in the model. The details of those events appear in this appendix.

The Key Leader Engagement Umpire listens for the StartKeyLeaderEngagement event. This component handles all events for key leaders and dynamic network entities in the model. Figure E.5 shows the StartKeyLeaderEngagement event in the KeyLeaderEngagementUmpire Component. A Key Leader Engagement event could result in information about a person, the network, or PAVE general knowledge. These represent the critical knowledge in the IW TWG. The engagement could also result in the key leader campaigning for the initiating actor. This has the effect of influencing agents in the population model when they hear a key leader "saying good or bad things" about an actor in the wargame. Providing an incentive to the key leader encourages the key leader to

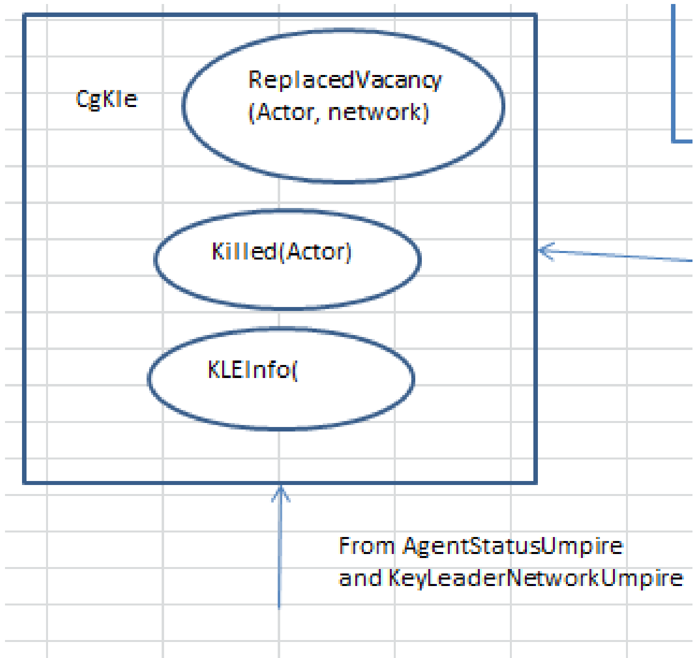

Figure E.4: KLE Component for Reporting Results to PAVE

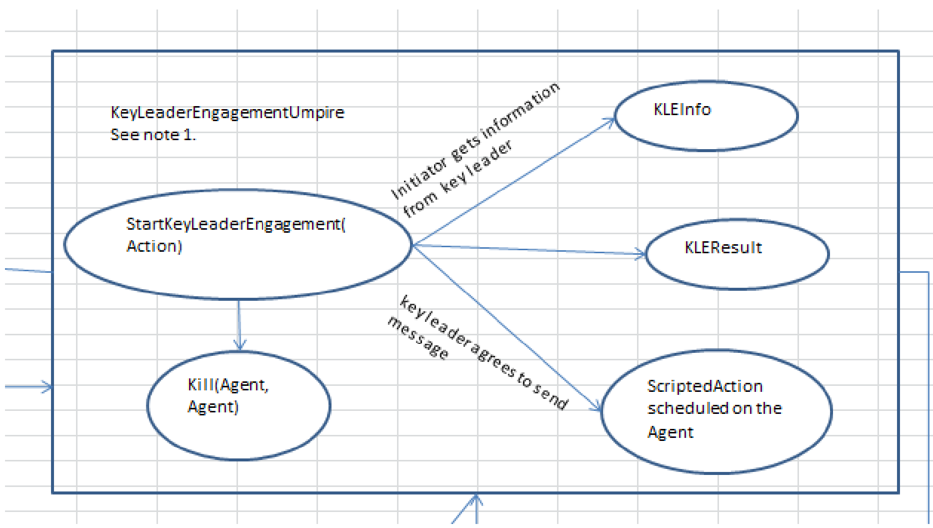

Figure E.5: Key Leader Engagement Umpire Component

cooperate. The next section (SIM Requirement UID 13.2.1) describes the incentive algorithm. A key leader can be killed or captured. That event triggers the Kill event in the Agent Status Umpire. Finally, the result of the Key Leader Engagement is the source for updating affinities between key leaders and initiators via the KLEResult node listening from the Affinity Network Umpire.

If a key leader is killed or captured during the wargame, SIM 2.0 handles the event with the AgentStatusUmpire Component. This umpire listens for a Kill event, triggering a call to the RemoveFromNetworks and Killed nodes (see Figure E.6). The Key Leader Network Umpire listens for a RemoveFromNetworks event to fire. With a random delay the KeyLeaderNetworkUmpire Component schedules a ReplaceVacancy event that accepts a network, role, subordinates and location as arguments. These variables "restructure" the network with a new key leader in the specified role. The source ReplacedVacancy node reports the change to the listening ReplacedVacancy node in the CgKle Component (Figure E.4).

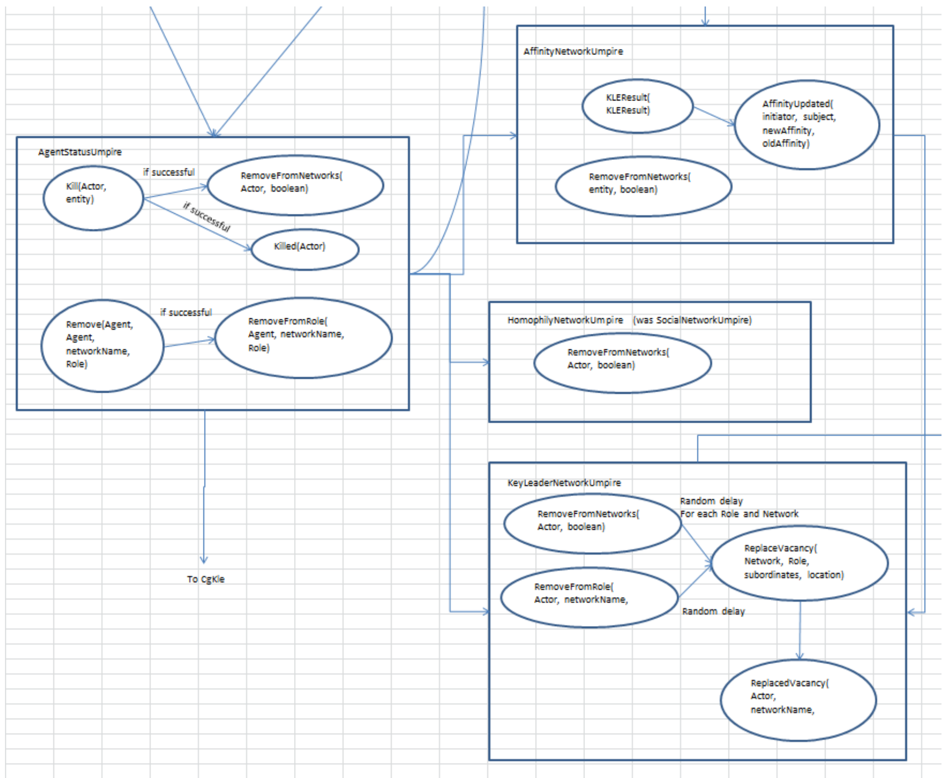

Figure E.6: Umpires for Managing the Static Networks

The AffinityNetworkUmpire and HomophilyNetworkUmpire Components in Figure E.6 update the affinity network. Similar to homophilly in the population model, affinity measures how much two key leaders like each other or another actor in the wargame. This helps determine a number of things such as how likely a key leader is to cooperate or whether an incentive is required to cooperate. It also assists in determining who replaces the key leader when removed or killed. The KLEResult event in the AffinityNetworkUmpire Component listens for the KLEResult event occuring via the Key Leader Engagement Umpire. In general, successful engagements result in a increase in one (1) affinity level and unsucessful engagement result in a decrease in one (1) affinity level. The IW TWG uses an affinity scale of 1 to 5 (levels) where a 1 represents significant distance or dislike between actors and 5 is akin to closeness or or key leaders affection.

The SocialNetworkBehaviorUmpire Component (Figure E.7) also listens for the ReceiveInformation event with a source in the Action Umpire. The knowledge arrives as a

Percept and allows the key leaders and light entities (micro key leaders in TWG 2011) to gain knowledge about SIM population model scripted actions via the GainSocialKnowledge node. The Social Network Behavior Umpire is also responsible for conducting behaviors that a scenario developer specifies in the scenario file in the "SocialNetworkBehavior" worksheet. This worksheet defines the start and stop distributions for the behaviors occuring because of RelationFormed and RelationshipBroken nodes. These relationships define the behaviors that a key leader or entity perform in the ConductBehavior node. Notice that the behavior node also has a self-scheduling arch because certain behaviors may schedule other subsequent behaviors in the model.

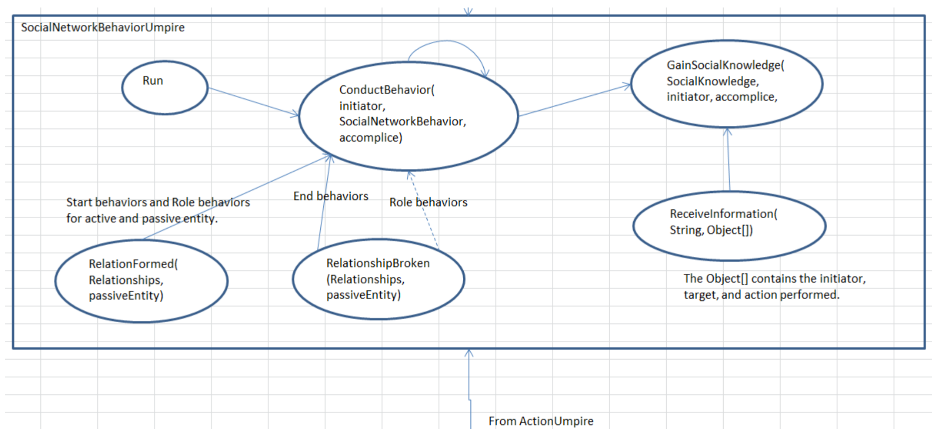

Figure E.7: Social Network Behavior Umpire Component

Finally, the DynamicNetworkUmpire Component controls the actions of the "Light Entities" in SIM 2.0. Light entities are agents that are not population agents, nor are they key leaders. They are just the "man on the street" that a TWG player can interview for information. They participate in dynamic networks that change over time during a scenario run. For instance, a man may get married, get a job, eat dinner, or learn that his neighbor is a bombmaker. If a player talked to this man, he might find out any of those three pieces of information (or many other possible ones). These networks get only as complicated as the scenario designer makes them using the KLE worksheets (See Appendix H). Figure E.8 depicts the event nodes contained in the Dynamic Network Umpire.

UID 13.2.1 The social network model shall provide an appropriate impact on actor capabilities within the wargame construct for all leaders in the network and all relevant actors of the wargame.

This requirement will be specific to a wargame. SIM 2.0 has the capability to model specific real-world key leaders, resulting in a higher classification level for the exercise. The use of actual names, demographics, influence and social-distance are capabilities of the KLE components of SIM. The discussion on the event graphs briefly touched on the idea of affinity; however, it is worth revisiting. Key leaders have a "bribe" value associated with them. This is essentially a corruption level and determines the incentives required to cooperate. The SIM implementation of this algorithm is simple:

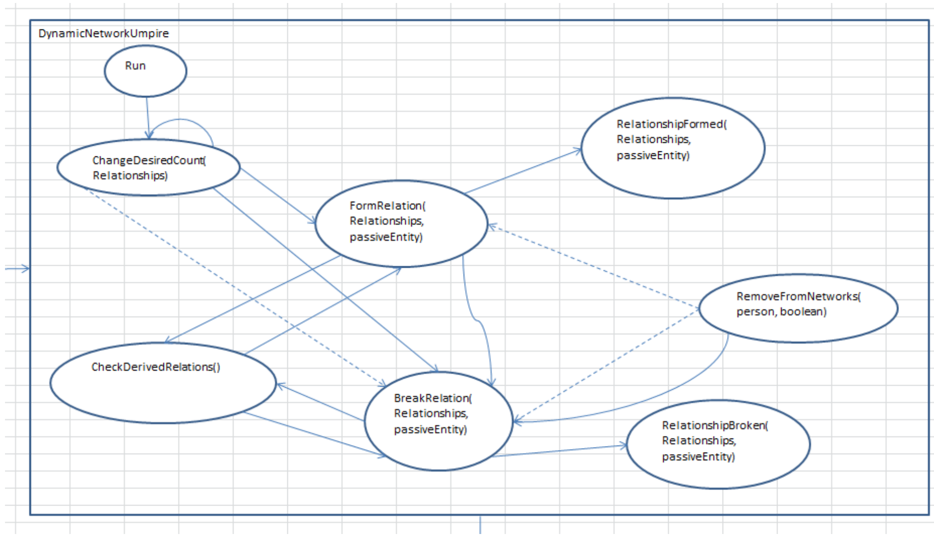

Figure E.8: Key Leader Perception Umpire Component

- If the affinity of the key leader toward another is Level 1, the key leader will refuse to cooperate and will be telling the truth about that refusal. He will also lie about having any critical knowledge.
- If the affinity level of the key leader is 5 toward another actor, the key leader will agree to a request and be telling the truth.
- If the key leader requires no incentive (or bribe), they will agree and be telling the truth.
- If an incentive is required but not offered, they will refuse to agree to the request and be telling the truth.
- If an incentive is required and the offered incentive is too small, the key leader will take the offer and lie about doing an action. This simulates the key leader having the action "take the money and run".
- If an incentive is required and the incentive is large enough, the key leader will agree and be telling the truth.
- If threatened, the key leader will always comply and be telling the truth; however, the affinity value will likely decrease between the key leader and the actor threatening the key leader.

Additionally, actors are able to able to gain intelligence through multiple means such as SIGINT, interviews, key leader engagements, or a Shura. The term Shura holds over from multiple TWG using an Afghanistan-based scenario. SIM 2.0 allows the scenario developer to call the meeting of several key leaders whatever they desire. Different boundaries built in the SIM 2.0 scenario file allow developers to call a meeting of key leaders in a variety of

AOs. For example, if company, battalion, brigade, and provincial boundaries reside in the SIM scenario file, any one of those areas could be used to request a key leader's presence. The leaders attending the Shura get reported back to PAVE for analysis.

Another key variable is the probability that a key leader's affinity toward another will change rationally. In addition to the affinity resulting from an engagement, affinity can also be updated randomly. This is important because, if all key leader decisions are rational, they will never move out of a Level 1 or Level 5 affinity state. See the User Guide in Appendix H for descriptions of the variables controlling other key leader behaviors.

UID 14.2.0 The social network model design/development lead shall coordinate with the IW TWG lead to ensure appropriate IW Enterprise organizations are aware of changes that impact their particular expertise

TRAC-MTRY coordinated with the IW TWG leads to ensure appropriate IW Enterprise organizations were aware of the changes that impact their particular expertise. Because SIM interfaces with PAVE, there was close coordination with PAVE to ensure inputs and outputs from the model comply with integration standards. After completing the development of SIM 2.0, the team determined that processes that were not automated in Nexus (SIM 1.0) were now produced in the SIM 2.0 model, thus enhancing the capability of the model. Nexus relied on contractor support from a computer scientist to interpret outputs from the model and directly input them into PAVE. A couple examples of outputs not produced by Nexus that are produced by SIM 2.0 are:

- Names of key leaders who attended a Shura, consultation, or town hall meeting (the name is now irrelevant and can be anything).
- Names of civilians interviewed (micro key leaders in TWG 2011).

Finally, all algorithms developed for SIM 2.0 that were not previously automated in Nexus were discussed with PAVE developers and vetted through the SIM Transition POC at TRAC-WSMR prior to implementation. If the algorithms do not accomplish TWG objectives, code changes would be minor to modify due to the simple, conditional nature of the algorithms.

E.2. ALTERNATIVE OAB METHOD

The SIM transition team sought to explore alternative means to represent OAB. The goal was to make results in the model more explainable. The development team contended that a five (5) state OAB was confusing. The use of Bayesian Belief Networks to represent OAB further clouded results from the model. In previous years SIM developers used BBN because it was easier to model OAB in the same manner as issue stances. However, the underlying survey data does not ask the population about attitudes toward various actors.

There was a loose association from survey data to attitudes. Furthermore, the five OAB states (NA, NP, N, PP, PA) are not easily understood since they are only probabilities of being in one of those states.

Leveraging the capabilities of SimKit, the team implemented a counter system where a positive action adds one to an agent's OAB toward an actor and a negative action subtracts one from and agent's OAB toward an actor. If the IW TWG team needs various states, these counters can be binned or thresholds could be set to represent different attitude states. Ultimately, presenting these scaled values to a player could produce a result that is better understood by a human player - high numbers are good, low numbers are bad.

This only changed one component of code within SIM - the LongTermMemory Component (Figure E.9). Appendix C provided an overview of Long Term Memory in SIM. The alternate version of SIM 2.0 only modifies the UpdateLongTermMemory node. Instead of sending values to the BBN, this node simply refers proper counter value established in the scenario file and increments the new OAB accordingly.

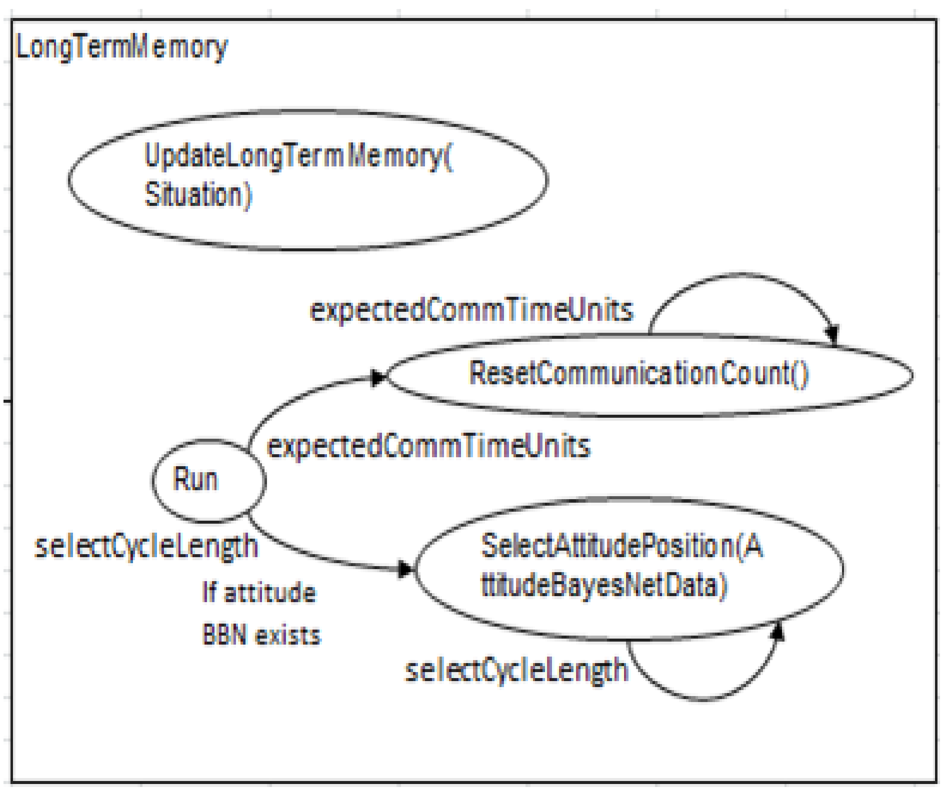

Figure E.9: Long Term Memory Component in SIM 3.0

APPENDIX F. SIM 2.0 TEST RESULTS SAMPLING

F.1. Overview

As outlined in the chapter on SIM 2.0, the first testing conducted on the 2.0 version were the baseline tests. Once developers validated that the population model results were the same for both versions, testing shifted to look for the recommended number of survey respondents needed to produce significant results in the model. The team recommends that population stereotypes contain around 100-survey respondents per case file.

Next, the team looked at the specific requirements from TRAC-WSMR. These focused on the impacts of events on population agents. Developers ran tests on both the TWG 2011 stereotypes, test stereotypes, and the Africa KDAE stereotypes. This section focuses on a few interesting cases from the Africa KDAE stereotypes.

Finally, Major Ong assisted with the extensive testing of the cognitive architecture covered in Appendix C. That testing was the foundation for his thesis work, included in Appendix F. The results of those tests enable the team to recommend only using Exploratory Learning in the cognitive architecture. See the User Guide in Appendix H for how to set the variables in the CognitiveArchitecture worksheet.

The SIM Transition team delivered all SIM 2.0 testing files to TRAC-WSMR on 21 September 2012. Those files included over 1400 input scenario files and 14,000 output files for analysis. Follow-on research would be easy to begin by using those files as a starting point. The remainder of this appendix highlights a few cases.

F.2. SPECIFIC TESTING REQUIREMENTS

CG UID 1.2.1 Provide a scale which indicates the impact for each quantity

This requirement sought to determine the impact of adjudicating multiple scripted events of the same type versus just one. The TRAC-WSMR team wanted to know if there was a threshold value where this impact changes. In conjunction with the DOE outlined in CG UID 1.2.0 the IW TWG requested that developers document the results.

Numerous tests were conducted on both TWG11 and KDAE data. For these tests, the research team was especially interested in the KDAE stereotypes because their case files contained a minimum of 300 survey respondents each. This was a richer set of data when compared to the TWG11 stereotype case files. Recall that case files with more evidence produce better outputs from SIM. Therefore, the KDAE data showed more interesting effects of the multiple actions and types of actions. Figure F.1 shows the results of 5 design points tested using the KDAE dataset. These design points were:

- DP_225: Baseline effects of an event that the population perceives as 100% positive.
- DP_243: Multiple, repetitive events per day.
- DP_261: Both positive and negative effects occurring each day.
- DP_279: Effects of an event that the population perceives at only 75% positive.
- DP_297: Effects viewed as 50% positive and 50% negative (50-50).

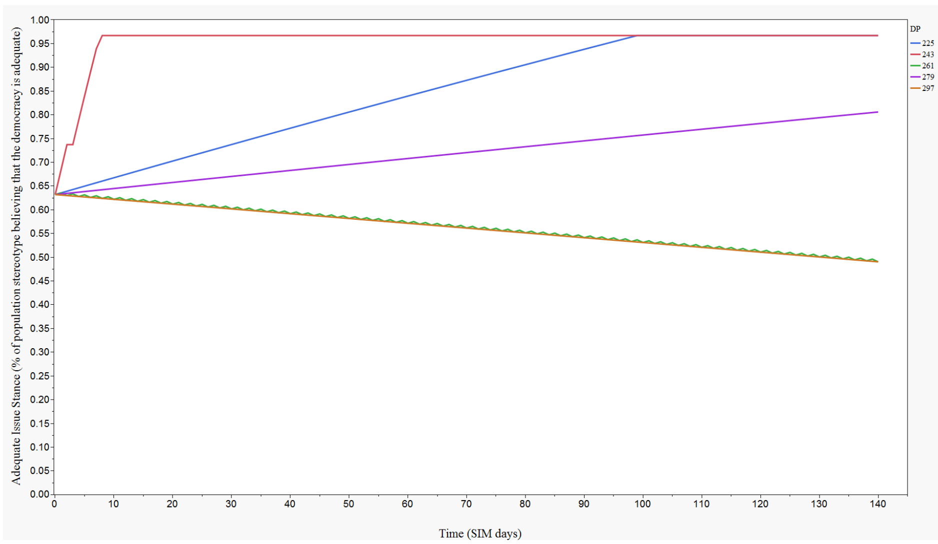

Figure F.1: Varying Effects and Numbers of Actions

The agent prototype used in these five design points is the F_C_O, Female Christians Over-30, detailed in the Testing section of the SIM 2.0 chapter. As with other baseline testing, DP_225 shows that a population agent will reach its highest issue stance value (% of opinion = adequate) around 100 SIM days, under the following conditions:

- Discount factor (λ) of 0.01.
- One event per day.
- Event effects = 100% positive.

To answer the requirement, the team chose several modifications to this baseline that ran multiple events per day. DP_243 highlights a design point where five (5) scenario events influencing the issue stance with 100% positive effects are scheduled per day. The red line in Figure F.1 shows that the agent reaches maximum issue stance at around 10-days. By the 10th day, 50 events have occured. The discount factor (0.01) has a lower effect on the resulting issue stance because only 10-days have passed, and the "memories" are still recent. This recentcy increases the effect of the event. The number of possible combinations for these variables gets combinatorially explosive to test. There are a numerous possible outcomes depending on the combination of discount factors, number of events per day, and the effect of each event per occurance. A calibration exercise prior to the next TWG will ensure that the right combination of variables is in the scenario file. To generalize, if a developer wanted to maximize the effects of events on an agent these three conditions would need attention:

- Use a low discount factor (λ). The team recommends 0.01 as a minimum; however, reducing it further will allow the effects of past events to last longer.
- Maximize the effect of the event on a population stereotype. SMEs should provide data for these effects, but the closer the effect is to 100% positive or negative, the faster the change will occur increase or decrease respectively.
- Maximize the number of events over short periods (number in a day or week for example). This is an easy way to do it, but it might not be realistic, depending on the event. For instance, you may not be able to award 10 projects to the same agent every day for a week. This simply does not replicate what happens in theater.

CG UID 2.2.0 Provide the results of a DOE on the SEs in order to determine the size of the effect

In conjunction with the DOE outlined in CG UID 1.2.0, the team tested the magnitude of scripted event's effects on the populations. The purpose of this testing was to document how effects altered population stereotype opinions. In DP_225 and DP_243, the effects were set to 100% positive, ensuring that SIM produced a maximum increase for a given BBN. For DP_279 and DP_297 those effects reduced down to 75% and 50% positive respectively. The purple line (DP_279) in Figure F.1 shows that changing the effect to 75% positive decreases the rate that population opinion increases. Finally, the amber line in Figure F.1 shows that 50-50 data causes the effect to converge near the 50% satisfaction line.

APPENDIX G. Major ONG'S THESIS: TESTING SIM 2.0

In order to simplify SIM 2.0, a detailed look at the population model's cognitive architecture was necessary. Specifically, the team suspected that the RPD and trust modules did not provide a significant effect on outputs from the model. By using both RPD and Exploratory Learning (EL), analysts require extra loggers to trace results from the model. Agent decisions that utilize RPD vary slightly from those employing an EL path. Furthermore, the use of these modules might create additional complexity without any benefit. These research questions required attention.

To answer these questions, the team recruited MAJ Chin Chuan "Chase" Ong from the Singapore Armed Forces. Major Ong approached TRAC-MTRY looking for a thesis topic for his Masters of Science (MS) in Modeling, Virtual Environments, and Simulation (MOVES). He sought a topic involving agent behaviors, and the SIM Transition team guided his research. Major Ong's results greatly assisted the team in making decisions about what components to recommend using and informed modifications to the final version SIM 2.0.

After testing over 30,000 replications in the cognitive architecture, the team is able to say with confidence that the use of both RPD and EL in the cognitive architecture provides no statistically significant advantage. Results from both modules end up with nearly the same end state or agent decision. Similarly, the only difference between using and not using the trust module is additional variance. It also provides no additional benefit to model outputs, and can create additional computational overhead - not formally measured as part of the testing plan. Therefore, the team recommends using EL only with the trust module turned off.

In addition to the evidence about RPD and EL, Major Ong's testing also uncovered a rare situation involving scenario event effects and population opinions. When LTC Caldwell and Major Ong examined the data, there was a design point where a population agent had a 99% adequate view civil security. While infrastructure was damaged, his opinion decreased. After a set amount of time, the team had a repair event for the damaged infrastructure. Even after this repair, the agent's opinion continued to decrease. When that test concluded, the agent opinion had decrease to a belief that civil security was only 40% adequate. After examining the code and reviewing the underlying equations for this calculation, the team verified that the model behaves properly. Future scenario designers must be aware that positive effects must be greater than the initial issue stance of agent stereotypes when the simulation begins. For instance, the agent described above with an initial civil security issue stance can only increase this issue stance if the effect is greater than 99%. Of course this is an extereme case, but it equates to the belief that anything less than perfection will disappoint the agent. Given the discount factor (λ) of 0.01, the agent will remember failure for 100 time units, and the TWG will most likely be over before the agent's opinion can ever be turned around.

Testing of SIM 2.0 focused on the cognitive architecture. This appendix contains the thesis dedicated to the conduct of that testing. Appendix F contains additional tests conducted on on the SIM 2.0 population model.

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ANALYSIS OF COGNITIVE ARCHITECTURE IN THE CULTURAL GEOGRAPHY MODEL

by

Chin Chuan Ong

September 2012

Thesis Advisor: Christian J. Darken
Second Reader: Jason Caldwell

This thesis was performed at the MOVES Institute
Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704–0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704–0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2012	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Analysis of Cognitive Architecture in the Cultural Geography Model			**5. FUNDING NUMBERS**
6. AUTHOR(S) Chin Chuan Ong			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943–5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ___N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

The Cultural Geography (CG) Model is a multi-agent discrete event simulation developed by TRAC-Monterey. It provides a framework to study the effects of operations in Irregular Warfare, by modeling behavior and interactions of populations. The model is based on social science theories; in particular, agent decision-making algorithms are built on Exploration Learning (EL) and Recognition-Primed Decision (RPD), and trust between entities is modeled to increase realism of interactions. This study analyzed the effects of these components on behavior and scenario outcome. It aimed to identify potential approaches for simplification of the model, and improve traceability and understanding of entity actions. The effect of using EL/RPD with/without trust was tested in basic stand-alone scenarios to assess its impact in isolation on entities' perception of civil security. Further testing also investigated the influence on entity behavior in the context of obtaining resources from infrastructure nodes. The findings indicated that choice of decision-making methods did not significantly change scenario outcome, but variance across replications was greater when both EL and RPD were used. Trust was found to delay the rate of change in population stance due to interactions, but did not affect overall outcome if given sufficient time to reach steady state.

14. SUBJECT TERMS Cultural Geography, Agent-Based Modeling, Human Behavior Modeling, Social Simulations, Reinforcement Learning, Recognition Primed Decision, Trust			**15. NUMBER OF PAGES** 79
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540–01–280–5500

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

**ANALYSIS OF COGNITIVE ARCHITECTURE
IN THE CULTURAL GEOGRAPHY MODEL**

Chin Chuan Ong
Major, Singapore Armed Forces
B.Sc., University of Illinois at Urbana-Champaign, 2005

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN
MODELING, VIRTUAL ENVIRONMENTS, AND SIMULATION (MOVES)**

from the

**NAVAL POSTGRADUATE SCHOOL
September 2012**

Author: Chin Chuan Ong

Approved by: Christian J. Darken
Thesis Advisor

Jason Caldwell
Second Reader

Christian J. Darken
Chair, MOVES Academic Committee

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Cultural Geography (CG) Model is a multi-agent discrete event simulation developed by TRAC-Monterey. It provides a framework to study the effects of operations in Irregular Warfare, by modeling behavior and interactions of populations. The model is based on social science theories; in particular, agent decision-making algorithms are built on Exploration Learning (EL) and Recognition-Primed Decision (RPD), and trust between entities is modeled to increase realism of interactions. This study analyzed the effects of these components on behavior and scenario outcome. It aimed to identify potential approaches for simplification of the model, and improve traceability and understanding of entity actions. The effect of using EL/RPD with/without trust was tested in basic stand-alone scenarios to assess its impact in isolation on entities' perception of civil security. Further testing also investigated the influence on entity behavior in the context of obtaining resources from infrastructure nodes. The findings indicated that choice of decision-making methods did not significantly change scenario outcome, but variance across replications was greater when both EL and RPD were used. Trust was found to delay the rate of change in population stance due to interactions, but did not affect overall outcome if given sufficient time to reach steady state.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AI	Artificial Intelligence
CF	Coalition Forces
CG	Cultural Geography
COIN	Counter-Insurgency
DoD	Department of Defense
DP	Design Point
EL	Exploration Learning
HA	Humanitarian Assistance
HSCB	Human Social and Cultural Behavior
IW	Irregular Warfare
M&S	Modeling and Simulation
MSCO	Modeling and Simulation Coordination Office
OAB	Observed Attitude & Behavior
OOTW	Operations Other Than War
PEO STRI	Program Executive Office for Simulation, Training & Instrumentation
RPD	Recognition Primed Decision
SSTR	Security, Stability, Transition and Reconstruction
TpB	Theory of Planned Behavior
TRAC-MTRY	TRADOC Analysis Center – Monterey
TRAC-WSMR	TRADOC Analysis Center – White Sands Missile Range
TRADOC	Training and Doctrine Command

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

This thesis would not have been possible without the many people who provided invaluable advice and assistance to me throughout the study. I would like to express my heartfelt gratitude for their patience, support and guidance.

My Thesis Advisor, Dr. Chris Darken, was the source of inspiration and motivation for my venture into this area of research. His teachings and in-depth knowledge of the field helped me significantly, and his advice over the course of testing and experimentation provided key insights and guidance that helped shaped this work.

The members of TRAC-Monterey provided invaluable support and coaching for me as we worked to understand the programming wizardry inside the Cultural Geography Model. My Second Reader, Lieutenant Colonel Jason Caldwell, provided me with a deeper understanding and sense of purpose of the work, and helped to broaden my perspective of the inner workings and applications of the Model in operational contexts. Mr. Harold Yamauchi, the programming guru and expert on the CG Model, was instrumental in the scenario creation, manipulation of design inputs and generation of data outputs. His hard work and expertise provided the foundation for all the testing and experimentation that we were able to achieve with the code. I am also grateful to Lieutenant Colonel Jonathan Alt, for giving me this valuable privilege and opportunity to work with TRAC-Monterey; and to MAJ Francisco Baez for his advice and knowledge sharing.

Last, but certainly not least, my sincere appreciation also goes to my wife and son for their support and understanding as they endured my long hours at school and late nights at home.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

In most modern defense-related ecosystems in the world today, Modeling and Simulation (M&S) has established itself as an effective and resource-efficient tool for training and preparation of military operations and other undertakings. The U.S. Department of Defense (DoD) Modeling & Simulation Coordination Office (MSCO) recognizes that “M&S is an enabler of warfighting capabilities. It helps to save lives, to save taxpayer dollars, and to improve operational readiness” (MSCO, 2012). Wargaming is one common application that allows planners and analysts to gain insight on likely combat outcomes, challenges and potential pitfalls, and other unintended consequences that cannot be captured by traditional analysis methods. In such applications, a key success factor is the ability to maintain an extensive database of fully or semi-automated entities that represent actors within the scenario, and these entities need to have the ability to portray the actions and behaviors of real life combatants. In combat-based models and simulations, relatively realistic portrayal of soldiers and units can be attained through reference to doctrine and tactics, which dictate rules for how the entities would move, interact and react to the situation (Pew & Mavor, 1998; U.S. Army PEO STRI, 2012).

However, in recent times, the spectrum of military operations has expanded tremendously, encompassing missions such as Counter-Insurgency (COIN), Security, Stability, Transition, and Reconstruction (SSTR) efforts, and Humanitarian Assistance (HA) missions. The shift away from conventional conflicts and armed, open fighting between states reflects the changing political and security landscape in the world today. With this, military leaders need the ability and tools to appreciate the planning considerations, courses of actions and challenges in such Operations Other Than War (OOTW) and Irregular Warfare (IW) situations (DoD, 2008; Ng, 2012). In these areas, the changes that military actions bring to the economy, society, and political situation in the area of operations are often the indicators of mission success (Joint Chiefs of Staff, 1995), and thus the ability to have prior understanding and insights on it is a crucial aspect that needs to be addressed.

Simulating the entities that exist in unconventional environments is complex, as the requirements and challenges for modeling non-combatants and non-traditional combatants such as insurgent fighters are very different. For example, the artificial intelligence (AI) driving the actions of a regular soldier agent may be scripted based on rules of engagement and small-unit tactics; however, the response of civilians in a crowd to the military presence would vary significantly, depending on their demographics, personal circumstances, and perception of the immediate and long-term situation around them.

In this respect, there is a well-recognized need to improve the modeling of realistic human social and cultural behavior (HSCB). This would allow greater fidelity and realism in simulations in the realm of non-lethal operations, where the ability to better captures the “softer” effects of military action and to understand the impact on the population and social structure would be an important contributor to success (Alt, Jackson, Hudak & Lieberman, 2009; Pew & Mavor 1998).

The Cultural Geography (CG) Model developed by the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center – Monterey (TRAC-MTRY) seeks to enhance existing DoD efforts to model the responses of populations and social networks to operations conducted by the military in OOTW and IW campaigns (Alt et al., 2009; TRAC-MTRY, 2009). The CG Model is a multi-agent, discrete event simulation implemented in Java that models populations as entities in a geographical area. The agents, or entities, in the model are based on demographic information defining parameters for their beliefs, attitudes towards other entities, and actions taken. The cognitive architecture module in the CG Model forms the foundation for the artificial intelligence of these entities, and is based on well-studied social theory, concepts and models, such as Icek Ajzen’s Theory of Planned Behavior (TpB), Bayesian Belief Networks, and representation of homophily and its effects on interactions between entities (Alt et al., 2009; Alt, 2010; Perkins, Pearman & Baez, n.d.).

B. PROBLEM STATEMENT

Currently, the Social Impact Module (SIM) Transition being undertaken by TRAC-MTRY and TRADOC Analysis Center – White Sands Missile Range (TRAC-WSMR) seeks to fine-tune the CG Model to increase its acceptability by the end-users (TRAC-WSMR). One of the possible areas of improvement is to simplify the artificial intelligence and agent behavior in the CG Model so that it is better understood during implementation and use.

The complexity of multi-agent systems like the CG Model, which has many linkages and interactions, makes it realistic as a representation of HSCB, but also increases the difficulty in tracing and understanding the behavior of agents in it, and thus the outcome of the simulation. This thesis seeks to investigate two key aspects in the cognitive architecture of the CG Model. First, the current decision-making process of the entities, which is based on two well-known models – Recognition Primed Decision making (RPD) and Reinforcement Learning (Baez et al. 2010; Ozcan, Alt & Darken, 2011); and second, the trust module within the CG Model, which provides an additional layer of realism (and with it, complexity) by simulating the effect of trust, or the lack of it, between entities in the scenario (Baez et al. 2010; Pollock, 2011).

These components in the cognitive architecture enhance the fidelity of the agent representation as the entities respond based on a greater range of possible options under the effects of the rules that they bring to the model. Individual studies have demonstrated statistically significant contributions of these components to the CG Model (Ozcan et al., 2011; Papadopoulos, 2010; Pollock, 2011). However, in terms of creating a believable, realistic entity that performs on par with end-user expectations, it is worthwhile to consider if similar entity behavior is attained by implementation of a simplified artificial intelligence, i.e., without contributions of varying decision-making methods, or the trust module. Essentially, an acceptable degree of realism in agent behavior needs to be incorporated in the model, while avoiding an overly prescriptive and cumbersome AI.

C. OBJECTIVES

This study thus aims to isolate and investigate the effects of the decision-making module and the trust module on the outcomes of agent behavior in several test scenarios. As part of the process, it would generate greater insight in tracing the actions of entities, and provide reasonable understanding of the behavior to improve the believability of the model. It would also identify possible areas for simplification in the cognitive architecture, to reduce complexity of the artificial intelligence in the model without compromising on realism.

This thesis seeks to address the following key questions:

1. What significant effects do the decision making and trust components provide in the existing cognitive architecture, and do these perform as expected / desired?

2. Can a simplification of the cognitive architecture provide a reasonable behavior for agents in the CG Model that is comparable with that of the existing framework?

It is envisioned that the experimental design, scenario development and data generated from the study will provide ample references for a better understanding of agent behavior in the CG Model. The study will thus facilitate fine-tuning of the CG Model (in particular the cognitive architecture) towards meeting the requirements of the end-users for the CG Model, as part of the Social Impact Module Transition.

D. METHODOLOGY

The initial thrust of this study was to isolate the components in the cognitive architecture that are of interest, and analyze their effects on outcomes and agent behaviors in a simple scenario with one, two or three entities. Only a small subset of the full capabilities of the CG Model were used, so as not to introduce excessive effects of external factors which were not being tested. In particular, the agent(s) were placed in a specific geographical location, together with an infrastructure node from which they periodically obtain consumable resources. Scripted actions were injected regularly to trigger responses and changes to entity behavior.

The single entity scenario serves to provide insight on the direct relation between the decision-making method and the entity's behavior and eventual outcome of the scenario. The two-entity scenario added the effect of trust, which would be visible in the form of communications between the two agents. The three-entity scenario furthered the analysis with the addition of another agent based on a distinctly different prototype than the original two. This third entity has a lesser degree of homophily to the other two, and thus the effects of trust and interactions with other agents or the environment would be dissimilar.

This initial analysis measured outcomes in terms of change in population stance, frequency of communications between entities, choice of decision-making method, and the effects of action selections on agent attitudes and stance. Overall, it provided insight on the direct effect that the decision methods and trust have on agent behavior and scenario outcome.

The results of the initial analysis provided the basis for the scenario development of the subsequent set of experiments. The scenario complexity was increased to create a more realistic depiction of a plausible, real-world situation. Six agents and 2 infrastructure nodes were placed in separate geographical locations, but within range of communicating with and reaching each other. Several revisions to the scenario parameters were tested in order to identify one that would best exploit and bring out the differences in the various configurations of the cognitive architecture. The final set up was one in which the infrastructure nodes were initially insufficient to supply the requirements of the agents, but a scripted action was introduced to occur after some time, to improve the state of infrastructure. The intent was to trigger changes in agent behavior after the occurrence of the scripted action, and identify the variations in response for agents reacting based on the different decision methods and effects of trust.

The data from the initial experimental runs and the various revisions leading up to the final run was analyzed to generate a statistical comparison of the outcomes from the basic decision making methods, with and without trust, compared to the existing cognitive architecture framework in which entities can choose between RPD and Reinforcement Learning, under the effects of trust.

II. OVERVIEW OF THE CULTURAL GEOGRAPHY MODEL

A. DEVELOPMENT

The ‘Representing Urban Cultural Geography’ project was conceptualized in 2006 as an initial prototype for a simulation of a population in a social network (Alt, 2010; Baez et al., 2010; TRAC-MTRY, 2009). Continued work over the next few years saw its development through various forms, with more components and features adding to the depth and complexity of the model, such as inclusion of entity actions (e.g., insurgent activity), representations of resources and infrastructure nodes, communications, and improvements to agent behavior modeling (Alt et al., 2009; Perkins et al., n.d.). The implementation also evolved from its earlier usage of the Pythagoras 2.0 agent based combat model (Ferris, 2008; Seitz, 2008) to its current form, which utilizes the SimKit Discrete Event Simulation in Java (Alt, 2010; Buss, 2011). A key feature of the model is its framework to allowing modules to ‘plug-and-play’ into the program (Alt et al., 2009), allowing flexibility and increased functionality. Two recent CG model developments are of relevance to this thesis—first, the use of a Reinforcement Learning based method for agent action selection (instead of a previous Bayesian network representation) (Yamauchi, 2012); and second, the implementation of a “trust” module that adds onto existing agent behavior. These two components are described in further detail later in this chapter.

As with all models, the intent for the CG Model is not to create a perfectly realistic representation of the world in order predict with absolute certainty what would happen in any given scenario—that would clearly be impossible to achieve. Rather, it provides a framework for analysts and planners to understand a situation and experiment with courses of action and alternatives to assess viability, possible outcomes, and potential pitfalls.

B. UNDERLYING CONCEPTS AND THEORIES

The representation of any real world process or phenomena as a model is intrinsically not an easy task. This is especially true in military and HSCB-based applications where there are a vast number of actors/objects, complex interactions, and lack of well-defined relationships and rules governing causes and effects. In order for the model to perform well, it must produce outputs that are rational and believable with respect to its intended purposes and areas of usage. In the field of HSCB modeling, this can be achieved by building the simulation based on theories in social science and psychology, along with clear understanding of the structure of organizations and demographics of populations being represented (Pew & Mavor, 1998). The CG Model is an example of this, as it is based on well-studied concepts and theories creating a rational and understandable framework for the representation and study of military operations in IW. A brief look at some of the underlying concepts and theories used in the CG Model follows.

1. Theory of Planned Behavior

Icek Ajzen's Theory of Planned Behavior serves as the basis for a core component in the CG Model. This theory attributes a person's intentions and behaviors to three key factors: his attitude towards the behavior, the subjective norms associated with that behavior, and his perceived behavioral control (Ajzen, 1985; Ajzen, 1991). *Attitude towards the behavior* describes the individual's own assessment of the behavior, for example if a person is in favor of always returning to the same provider to obtain a particular resource or commodity. The *subjective norm* brings out the social dimension as it represents the degree to which there is external influence (such as from peers and the community) towards the behavior, for example if a person's local community utilizes a particular other resource provider and pressures him to do the same. The *perceived behavioral control* gives a measure of how easily the individual believes he can carry out the particular behavior, for example if he has the ability to make the switch to a new resource provider. Ajzen postulates that the combination of these three independent factors determines the individual's intention to behave in a particular fashion, and that the intention and perceived behavioral control in turn determine the actual behavior adopted (Figure 1).

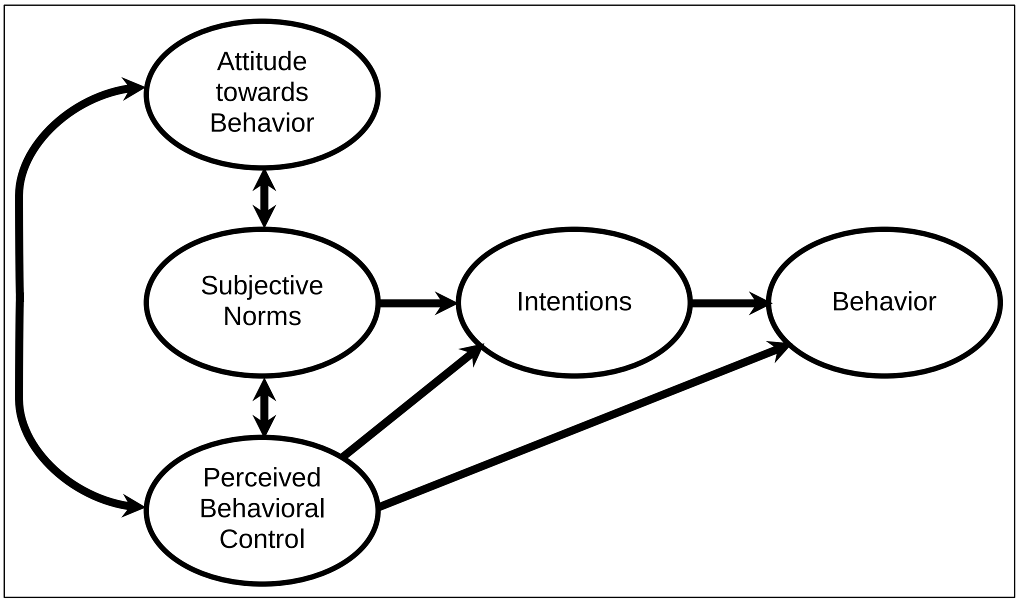

Figure 1. Theory of Planned Behavior (From Ajzen, 1991).

Within the CG Model, these three factors apply to each entity in any given scenario, and are quantified to derive a value for each behavior that the agent may choose. The *attitude towards behavior* is influenced by the agent's demographic stereotype and perception of issues relating to that behavior, the *subjective norm* is determined from the behavior of neighboring agents, and the *perceived behavioral control* is determined from the degree that a selected behavior brings about the agent's desired effect (essentially, a measure of success of behavior choices). User-defined weights are applied to the calculated values of the three factors, and the weighted sum is then used the measure of reward gained from a particular behavior (Yamauchi, 2012), as shown in the formula:

$$v_r = w_A v_A + w_N v_N + w_C v_C$$

where

$v_r = reward\ value\ of\ behavior$

$w_A = weight\ of\ Attitude\ towards\ Behavior$

$w_N = weight\ of\ Subjective\ Norms$

$w_N = weight\ of\ Perceived\ Behavioral\ Control$

$v_A = value\ of\ Attitude\ towards\ Behavior$

$v_N = value\ of\ Subjective\ Norms$

$v_C = value\ of\ Perceived\ Behavioral\ Control$

2. Narrative Paradigm

The Narrative Paradigm (Fisher, 1984) provides the logic through which populations in a real-world area of interest are converted to agent representations in the CG Model. Fisher's work proposes that an individual's experiences in life form a collection of narratives that describe his culture and character, shapes his perspective of the world, and affects how he responds to events and interacts with others around him. As such, the narrative account can be used as a comprehensive and credible data set for the purposes of classifying population as different entities, each with its own unique demographic traits and stereotypes for responding to the environment. The CG Model directly implements this by having each entity represent a subset of the population in the area of interest, with the entities ranging from a single individual, to a small group or entire community. Input parameters that are required by the simulation to adjudicate interactions and behavior of agents are then derived from their respective narratives and demographic traits. Table 1 lists the social dimensions and categories for the Afghan Helmand Province data, which was used in this study (Hudak & Baez, n.d.).

Social Dimension	Categories
Family Status	Inherited
	Achieved
	Unemployed
Ethno-Tribal Affiliation	Pro-Government
	Passive
	Marginalized
Disposition	Urban
	Rural
Political Affiliation	Fundamentalist
	Moderate
	Secular
Age	Military Age Male
	SpinGiri[1]

Table 1. Social Dimensions & Categories in Helmand Province Population Narratives (From Hudak & Baez, n.d.)

An entity stereotype is determined by a combination of traits from the list above that forms its demographic profile, along with the initial data of the entity's attitude and beliefs towards other entities and stance on pertinent issues in the scenario, such as the adequacy of Civil Security in the province.

3. Homophily

The concept of homophily is closely tied to modeling interactions between different population groups in the CG Model. Homophily refers to the similarity between individuals and affects the likelihood that two parties would associate and interact with each other. Its effect is most visible in social network contexts, where similarities and differences in demographic traits and social factors have a pronounced effect on the number and extent of links between people (McPherson, Smith-Lovin & Cook, 2001). This suggests that the effects of

[1] "Spin Giri" is a term referring to senior males who are typically past the traditional warrior/military age, are influential and likely to be local decision makers or hold other positions of tribal leadership (Hudak & Baez, n.d.).

homophily can significantly influence the behaviors of individuals and outcomes of scenarios.

In the CG Model, similarity between entities is determined in accordance with this concept of homophily. The stereotypes (i.e., demographic traits) and geographical proximity of entities are the main factors in the computation, which generates a *homophily link weight* value for each entity pair in the scenario. This link weight is utilized to determine likelihood of communication between the entities, and would affect the sharing of information percepts in the scenario (Alt et al., 2009).

4. Decision Making and Learning

The process of making decisions is a key aspect of human behavior that is modeled in the CG Model. Two main concepts are implemented in the action selection component of the cognitive architecture—the Reinforcement Learning model and the Recognition Primed Decision model.

a. Reinforcement Learning

Reinforcement Learning is a technique of machine learning that determines how agents should act in a situation to generate an optimal overall outcome, based on a specified measure of the estimated value of each possible action. In a given environment, an agent receives information percepts that determine which state it is in, and selects an action from a set of possible options (Russell & Norvig, 2010). The resultant transition to a new state is assessed based on a predefined set of rules, typically in the form of some immediate reward given to the agent. By determining the overall value of each state-action pair (i.e., of choosing a particular action when in a particular state), the agent can make decisions that will allow it to gain the most benefit, or expected utility. The Q-Learning algorithm (Watkins, 1989; Watkins & Dayan, 1992) is implemented in the CG Model. This technique allows the agent to compute and iteratively update the expected utility of actions based solely on the rewards received from them, and not requiring the environment to be explicitly known, which is well suited for typical scenarios in the CG Model.

Reinforcement Learning provides agents with the ability to adapt well in new situations, where there is a strong impetus for behavior to *explore* possible options and identify the overall optimal course of action. Over time, the value of exploring diminishes as most or all options would have been covered, and the agent can shift its behavior to *exploit* only those actions with high expected utilities. This idea of trade-off exploration and exploitation is well studied; in particular, Ozcan et al. (2011) investigated several techniques for driving agent behavior in the CG model to optimize the balance between them. The action selection process in the CG Model is based on the Softmax method using a Boltzmann distribution, as depicted by the equation:

$$P_i = \frac{e^{E_i/t}}{\sum_j e^{E_j/t}}$$

where

$P_i = Probability\ for\ selecting\ action\ i$

$E_i = Expected\ Utility\ of\ action\ i$

$t = Temperature$

The probability of selected a particular action is determined by its expected utility (as compared to that of other actions) as well as a temperature parameter, which influences the exploration-exploitation balance (Baez et al., 2010; Yamauchi, 2012). Thus, an action has a higher probability of being chosen than any other action that has a lower expected utility. In addition, as temperature decreases from its initial value towards zero, the probability of choosing the action with the highest expected utility tends towards one, which gives rise to a purely exploitative behavior.

In the context of the CG Model's cognitive architecture, the Exploration Learning (EL) method[2] within the action selection module implements this generic reinforcement learning algorithm in accordance with the process developed by Papadopoulos (2010). Papadopoulos identified that the utility-based reinforcement learner was able to function well in the context of selecting the most appropriate action to drive a specified outcome, depending on the settings for parameters such as the initial temperature for the Boltzmann Distribution, learning rate and discount factor of the Q-Learning algorithm and initial expected utilities of actions. These parameters are user-defined values specific to each agent in the scenario, and thus grant the CG Model great flexibility for customization of agent reinforcement learning behavior.

b. Recognition Primed Decision Model

Recognition Primed Decision is a well-known model for naturalistic decision-making propounded by Klein (1989). It describes the theoretical process by which humans are able to make rapid assessment of a situation and come to a good decision without the need for extensive analysis to identify alternatives and then to compare the possible options to deal with the scenario. Klein noted that such behavior could be observed in experienced decision-makers in operational settings, such as firefighter commanders and small unit leaders in the military (Klein, Calderwood & Clinton-Cirocco, 1986; Klein, 1989; Klein, 1999). The RPD model suggests that in complex or time-constrained situations, such experts in their field are able to recognize cues and patterns that allow them to identify an effective course of action quickly, and that this technique would surpass a more deliberate, analytical approach in dealing with the situation.

In the CG Model, the implementation of the RPD model is largely based on the reinforcement learning technique described earlier. During a simulation run, agents will initially utilize the EL method and choose actions in an

[2] The term "EL" is used here-on to denote the *implementation* of the reinforcement learning algorithm in the CG model. This maintains consistency with the method name used in the CG Model source code and concept diagrams.

almost random manner (assuming that the initial expected utilities of actions are fairly similar). The number of times that the agent has taken any particular action is recorded, and compared to a user-defined minimum threshold, which dictates the number of times that an agent needs to perform each possible action before it is deemed to have sufficient experience. Upon reaching this threshold, the agent will adopt the RPD method of action selection, in which the action with the highest expected utility will always be selected during the decision making process (Yamauchi, 2012).

There are limitations in such an implementation—in particular, it does not capture some characteristics of the RPD model as described by Klein. The implementation in the CG Model is essentially a 'greedy' approach of reinforcement learning, where an agent has had the ability to explore various options in the environment before making a decision. In contrast, for a pure RPD approach, this benefit of time and knowledge of action-reward history may not be available to the decision maker. Rather, an agent having made no prior action selections in a particular scenario or environment (and thus having no corresponding estimates of expected utilities of possible actions) would have to decide its course of action based on the limited set of percepts it receives, using other knowledge such as its prior experience and long term memory. In addition, a decision maker in the RPD model would possess the pre-requisite ability to recognize changes in situation and discard previously adopted courses of action that are no longer effective (Klein, 1989; Klein, 1999). The implemented method does not allow agents to have such versatility, thus limiting their 'expertise' to situations that are relatively static. Significant changes in a scenario would likely not result in a responsive change of agent behavior once it has adopted RPD, as it would require time for the expected utility of the selected action to drop (until it is no longer the 'best' action) before the agent chooses another action.

The RPD model suggests that complex underlying thought processes are involved. For example, picking up cues from a situation (that may only be perceptible to experts but not novices); recognizing patterns that resemble previously encountered situations; and rapid mental run-through of a possible action to determine its feasibility on its own (as opposed to comparing it against a set of alternatives). These processes cannot be easily incorporated into the existing cognitive architecture of the CG Model, as it could require extensive restructuring of the framework, such as distinguishing between percepts received by expert entities (versus novice entities). This would better represent the significant differences in the performance characteristics of experts in a particular field (Proctor & Zandt, 2008), and thus better suit the implementation of a RPD model. Furthermore, it could require the introduction of larger and more complex long-term memory structures that can be used to compare past scenarios and experiences of an agent against a new situation in which it has limited percepts and situational awareness. Given the constraints in the cognitive architecture framework and the limitations of the current implementation, the RPD method in the CG model is an imperfect but necessary substitute for an actual RPD model.

C. COGNITIVE ARCHITECTURE MODULE

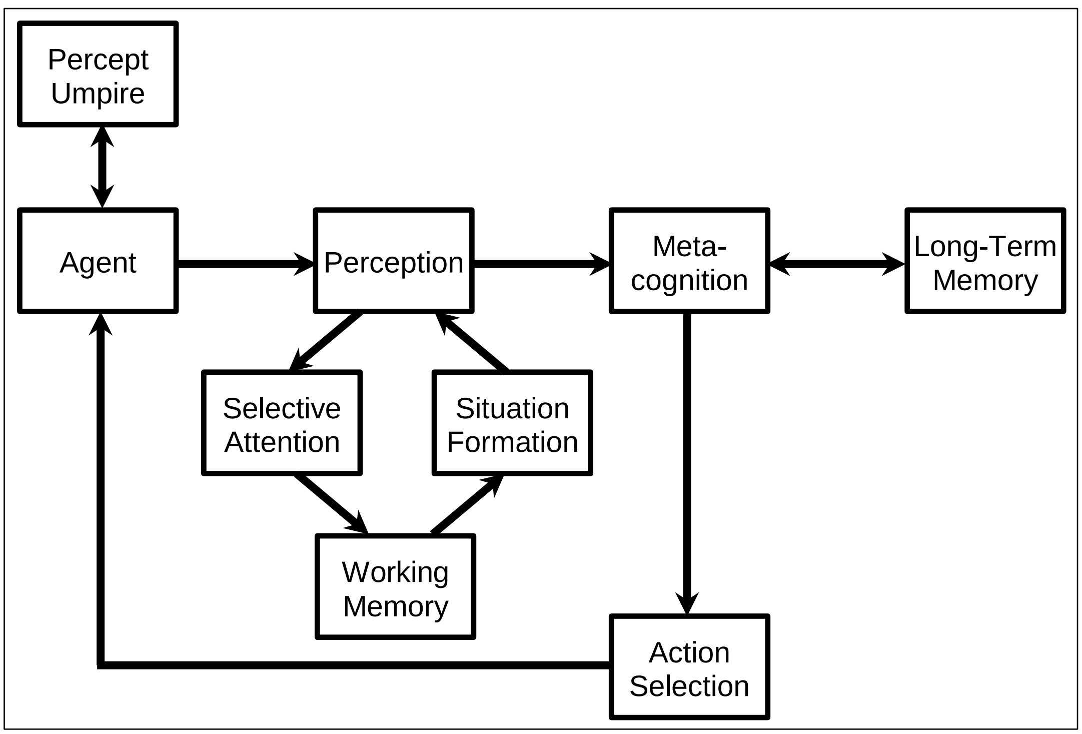

Figure 2. Cognitive Architecture Components (From Yamauchi, 2012).

The main components of the cognitive architecture module are shown in Figure 2, and their functions are described below.

1. Percept Umpire

The Percept Umpire acts as the ‘sensor’ for agents in the CG model. It receives information from the environment and entities in the model, such as changes to the state of infrastructure nodes, actions carried out by entities and consumption of resources by entities. These are scheduled as percept arrival events for the entities that are supposed to receive them.

2. Agent Object

The Agent component manages the actual state of entities in the CG Model, and is responsible for scheduling events such as performing actions, consuming resources and passing on percepts to the environment and other entities (through the percept umpire).

3. Perception, Attention, Working Memory and Situation Formation

When the entity receives percepts via the percept umpire, the Perception component of its cognitive architecture manages this incoming information, such as monitoring if the agent has the selective attention capacity to accept the information; checking the percept for relevancy and storing it in the working memory of the agent; and using this to schedule the meta-cognition events which are the precursors to the entity’s decision making and action selection processes.

4. Meta-Cognition and Long-Term Memory

The meta-cognition and long-term memory components represent the entity’s comprehension and assessment of its situation. Key events such as changes in attitude towards other entities or issues are scheduled within these components. The outcome of these stages is to determine possible courses of action for the entity based on the external situation and its internal motivations, attitudes and beliefs, and schedule the event for the agent to select a decision-making method and then make a decision.

5. Action Selection

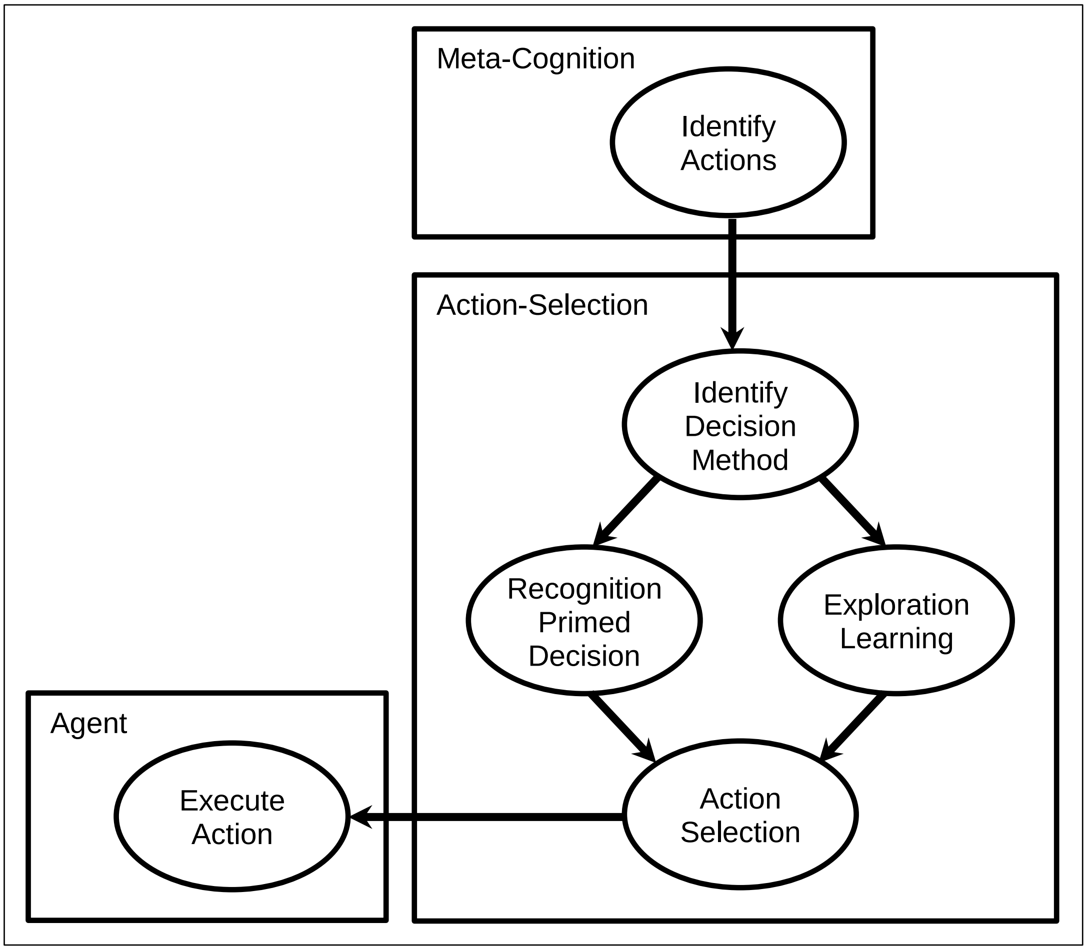

Figure 3. Action Selection Process (From Yamauchi, 2012).

The action selection component (Figure 3) is the main aspect of the cognitive architecture that is studied in this thesis. The process begins with the list of actions received from the meta-cognition component, which determines the type of decision-making method to use—either Exploration Learning (EL) or Recognition Primed Decision (RPD). The event to determine this takes into account the number of times that each possible action has been performed in the past, with the lowest count deemed as the entity's experience. This gives a simple and effective check to assess if the agent has sufficiently sampled all possible state-action pairs to build an accurate estimate of their expected utilities. Either the RPD method or EL method is scheduled, depending on whether the minimum experience has been reached. Thus, the minimum experience threshold parameter (pre-defined by the user) directly controls the amount of exploration that entities are allowed before they settle in the ‘greedy’ RPD mode. Once the decision-making method has been determined, the entity selects the appropriate action based on the probabilities evaluated from the range of expected utilities (or, simply selects the action with the highest expected utility in the case of RPD), and schedules the event for it to be carried out.

The action selection process also includes methods to initiate other scheduled events such as scripted behavioral actions and the cancellation of existing actions if necessary. These are methods are not investigated for the purposes of this study.

6. Communication and Effects of Trust

The CG Model simulates the interaction of entities and passing of information as communication actions taken by agents, such as the sending and receipt of percepts between them. This interaction influences the decisions and actions of entities, as it influences the parameters that are passed through their planned behavior process, in particular their *attitudes towards behaviors* and the effect of *subjective norms*. Pollock (2011) developed algorithms for representing trust between entities in a social structure, which aimed to capture additional facets of the relationships and effect of communications between agents.

Scenario designers initialize entities with parameters that determine their frequency of communication with other agents, while their similarity to others (as expressed through the homophily link weights) influences who they choose to communicate with. The trust filter implemented by Pollock interjected a check into the communication process that measures the level of trust between two communicating agents. The parameters for initial trust and changes to trust levels during run-time are defined in the scenario set up. With this trust filter, entities will still receive, but not accept or process, information received from agents that do not satisfy minimum trust requirements (Yamauchi, 2012). Pollock (2011) noted that inclusion of trust into the interactions reduced the rate at which agent changed their beliefs to align themselves with others. This study will look further at the effect on the overall scenario outcomes, as well as possible influences in conjunction with the choice of decision-making method.

III. ANALYSIS OF DECISION METHOD AND TRUST EFFECTS

A. DESIGN PARAMETERS

The experimental set up was designed to test two main aspects in the cognitive architecture of the CG Model—the decision making method, and the effect of trust. This corresponds to the following six basic test configurations:

1. Recognition Primed Decision only, without the effects of trust.

2. Recognition Primed Decision only, with the effects of trust.

3. Exploration Learning only, without the effects of trust.

4. Exploration Learning only, with the effects of trust.

5. Selection of either Recognition Primed Decision or Exploration Learning, without the effects of trust.

6. Selection of either Recognition Primed Decision or Exploration Learning, with the effects of trust. This is the typical configuration that is used in the current CG Model.

The tests were conducted using the Tactical Wargame 2011 (Revision 1160) version of the CG Model, as well as a modified variant of this version for the RPD only cases, in which the EL method of action selection was disabled. Entities in the RPD only variant would consistently choose the action that has the highest expected utility. This implementation serves to remove or reduce the ability of agents to gradually explore possible options and iteratively evaluate the expected utilities of all actions, and thus mimics human behavior in accordance with Klein's model of RPD. However, it is still limited by the inability to duplicate the process of rapidly assessing a new situation and selecting an effective solution based on one's expertise. The test configurations in which entities only use the Exploration Learning method were created by implementing a very high minimum experience threshold of 1000. This meant that the agents were forced to consistently choose the EL method over RPD, as the scenario run times were not long enough for them to have attempted all possible actions at least 1000 times each. The baseline configuration where entities could adopt either RPD or EL was set up using a minimum experience threshold of five.

The trust effects were tested by disabling the calculations of trust in code for the relevant configurations. The result of this is to prevent entities from performing checks that would disregard communications from senders whom they did not trust.

All other input parameters that are required for proper functioning of the cognitive architecture (in particular, for the Q-Learning Algorithm, Softmax algorithm, behavior utility calculations and trust module) were kept constant across the 6 test configurations. Table 2 summarizes the key input parameter settings that were used.

Configuration	**1**	**2**	**3**	**4**	**5**	**6**
Decision Method Settings	EL method disabled		Min Experience Threshold = 1000		Min Experience Threshold = 5	
Trust Filter Settings	Off	On	Off	On	Off	On
Reinforcement Learning Parameters	Initial Temperature = 0.1 Discount Factor, Lambda (λ) = 0.01 or 0.1 (see below)					
Behavior Parameters	Weight of Attitude towards Behavior = 0.3 Weight of Subjective Norms = 0.3 Weight of Perceived Behavioral Control = 0.3					
Trust Parameters[3]	Default Trust = 0.5 Learning Rate = 0.8 Discount Factor = 0.3 Trust Temperature = 0.5					

Table 2. Input Parameters for six Basic Test Configurations.

[3] Pollock (2011) provides a detailed investigation of the effects of these parameters, which are used in the algorithms pertaining to the reinforcement learning of trust, and affect the rate at which entities' trust fluctuate during the scenario runs.

In addition to the six test configurations, three other factors were varied for the initial set of tests: (1) the Reinforcement Learning Discount Factor, Lambda (λ), (2) the effect of scripted actions taking place during the scenario, and (3) the initial belief and issue stance of entities in the scenario. These factors had earlier been studied as part of the ongoing testing and evaluation by TRAC-MTRY, and were incorporated in the initial run to extend the number of data points over which the basic configurations could be tested.

The reinforcement learning discount factor (λ) was tested at two levels (0.01 and 0.1). The former corresponds to behavior that favors short term rewards, as the value of rewards (i.e., their contribution to expected utility of an action) diminishes more rapidly with time, while the latter corresponds to behavior that favors longer term rewards.

The effect of scripted actions was set to be either positive or negative, while the initial belief and issue stance of entities was varied over 14 possible cases. Further elaboration of these two factors is provided in the next section.

B. TEST SCENARIO

For the purposes of the initial run, a simplistic test scenario was used in order to minimize interactions from other components in the CG Model, and allow the effects of the test configurations to be isolated. This test scenario was developed based on the Helmand Province Case Study developed by the IW Study Team at TRAC-MTRY (Baez et al., 2010; Hudak & Baez, n.d.). The study encompassed several districts in the province, and generated a significant amount of data and analysis pertaining to the population demographics and their views three key issues—security, infrastructure and governance. It serves as a well-documented starting point for the purpose of scenario creation in the CG Model by providing rich datasets that facilitate the development and selection of initial parameters, and has been used in several other studies conducted by TRAC-MTRY (Alt et al., 2009; Perkins et al., n.d.; Wiedemann, 2010).

In the test scenario, two identical infrastructure nodes were sited within the area of operation, and constantly provide a consumable resource (electricity) to either one, two or three agents in the scenario. These agents consume the resource at a constant rate, and may carry out the action of visiting the infrastructure nodes to restock their supply as dictated by their behavior.

In the 1-agent and 2-agent cases, the entity prototype was assigned the social dimensions of *Inherited* family status, *Pro-Government* ethno-tribal affiliation, *Urban* disposition, *Secular* political affiliation, and *Spin Giri* age group. This is a typical entity used in the CG Model, abbreviated as I_P_U_S_Sp. In the 3-agent cases, the third entity was assigned social dimensions that were dissimilar from I_P_U_S_Sp – Unemployed, Passive, Rural, and Moderate, and Military age (Un_Pa_R_M_Ma). This distinction reduces the degree of homophily between the third agent and the other entities, to lower their homophily link weights and bring out any differences in behavior due to the effects of trust.

The population stance on the issue of civil security was used as the primary measure of scenario outcome, and the overall effects of the test parameters. This issue stance represents the percentage of the population (more precisely, of the groups represented by each entity in the scenario) who perceive that the level of civil security in the province is adequate. This issue stance is affected by many factors in the model, such as the beliefs of a particular demographic group as determined by their population narrative (e.g., the belief that Coalition Forces are not trustworthy or that the area is not a safe). Also, the occurrence of events during run-time (such as Insurgent or CF activity) and information passed on from other entities during the scenario (Yamauchi, 2012) are significant influences on the issue stance..

In addition, each entity possesses a set of attitudes and behaviors towards certain groups or issues. This is quantified as an *observed attitude and behavior* (OAB), which translates to one of five levels—positive-active (PA), positive-passive (PP), neutral (N), negative-passive (NP), and negative-active (NA). The OAB of interest to this study is that pertaining to the entities' perception of CF

(*OABtowardsCF*). An entity that is positively inclined towards CF but does not actively carry out actions in support of them would have an *OABtowardsCF* value that falls in the range corresponding to positive-passive; an entity that is negatively inclined and is likely to choose actions such as aiding insurgents would have an *OABtowardsCF* in the level of negative-active (Yamauchi, 2012).

Seven different settings were used for the initial belief and issue stance ("casefiles") of the entities in the test scenario. These correspond a combination of high/low extremes and mid-point levels for these two parameter (issue stance on civil security and *OABtowardsCF)*, and are shown in the summary of design factors/levels in Table 3.

In addition, a periodic scripted action was implemented in the scenario, representing the operation of Coalition Forces (CF) within the area that is visible to the agent(s). This scripted action was programmed to have a positive effect on the population stance on the issue of civil security in the area for half of the test cases, and a negative effect for the rest.

A final parameter that was varied was the size of dataset used as input parameters. This represents the sample size of the data collection process that is used to generate the entity stereotypes based on the population narratives. A setting of either 1000 or 100 respondents was used, to verify that reduction of the sample size would not have an impact on the consistency of results or overall outcome of scenario.

With 6 basic configurations – three settings for decision method (RPD / EL / Both) times two settings for trust (ON / OFF) – two settings for discount factor, seven settings for initial belief and stance, two settings for scripted action effect, and two settings for data sample size, a total of 336 design points were generated for the 2- and 3-agent scenarios. One hundred sixty-eight design points were generated for the 1-agent scenarios (as the trust-ON setting is irrelevant in this context). This created a total of 840 design points for the initial run. Table 3 provides a summary of the factors and settings used.

Factor	Number of Settings	Settings
Number of Agents	3	1-Agent: I_P_U_S_Sp_1
		2-Agent: I_P_U_S_Sp_1, I_P_U_S_Sp_2
		3-Agent: I_P_U_S_Sp_1, I_P_U_S_Sp_2, Un_Pa_R_M_Ma_1
Decision Method	3	RPD Only
		EL Only
		Both
Trust	2	On (Not applicable in 1-Agent case)
		Off
Discount Factor	2	0.1
		0.01
Scripted Action Effect	2	Positive
		Negative
Dataset Sample Size	2	100 Respondents
		1000 Respondents
Initial Casefile	7	Civil Security Stance: 100% Adequate OAB towards CF: 99% PA, 1% NA
		Civil Security Stance: 99% Adequate OAB towards CF: 99% PA, 1% NA
		Civil Security Stance: 50% Adequate OAB towards CF: 99% PA, 1% NA
		Civil Security Stance: 50% Adequate OAB towards CF: 50% PA, 50% NA
		Civil Security Stance: 50% Adequate OAB towards CF: 1% PA, 99% NA
		Civil Security Stance: 1% Adequate OAB towards CF: 1% PA, 99% NA
		Civil Security Stance: 0% Adequate OAB towards CF: 0% PA, 100% NA

Table 3. Summary of Design Factors and Settings.

Each design point was replicated 30 times, using a fixed set of 30 random seeds for all design points. The scenario was allowed to run for 140 days (simulation time), to allow sufficient time for trends in the performance measure to be seen, and steady state outcome to be observed.

C. OUTPUT PROCESSING

Dataloggers in the CG Model were used to record pertinent data from the scenario replications during run-time. The key parameters that were measured are shown in Table 4.

Parameter	Datalogger(s) Used	Description
Civil Security Issue Stance	PositionChange-PeriodicDataLogger PositionChange-DataLogger	Each entity's stance on the issue of civil security was recorded on a daily basis to monitor its change over time. Specific events (e.g. receipt of communications) resulting in changes in stance were also recorded.
Choice of Decision Method and Actions	DecisionMethod-DataLogger SelectAction-DataLogger	Every occurrence of the event where an entity chooses a particular decision method (RPD or EL) was logged, along with the entity's level of experience at that time. The action selected as a result of the decision method used, and the expected utility of the action, were also recorded.
Communications	CommCount-DataLogger Communication-DataLogger	All communication events between entities were recorded to keep count of the total number received by each entity, and the number that the entity rejected (due to the trust effects) The trust level between the two entities involved in each communication event was also logged.
Degree of Homophily between Entities	HomophilyNetwork-DataLogger	The homophily link weights between any 2 entities in the scenario were recorded periodically (every 30 days).
OAB	PositionChange-DataLogger	The OAB of entities towards CF was recorded for each event that triggered any changes in the level. This log measured the percentage of the population represented by each entity that fall into each of the 5 levels of OAB. This parameter was tracked for the purpose of cross-referencing with the issue stance, but not used directly as a measure of scenario outcome.[4]

Table 4. Description of Key Parameters Measured.

[4] Prior testing and evaluation by TRAC-MTRY had suggested that issue stances were more appropriate and better understood as measures of changes and outcomes in scenarios, compared to OABs. (J. Caldwell & H. Yamauchi, personal communication, July 2012).

Due to the large volume of data generated[5], a combination of manual and batch-file processing methods were used to organize the outputs into similar dataset groupings. These were further processed with SAS Institute's JMP Pro (version 10) statistical software to consolidate datapoints into relevant parameters, such as mean and variance across replications, trends over time periods in the scenario, and differences between entities and initial casefiles. JMP was also used for the analysis of the data and generation of plots.

D. RESULTS – SINGLE AGENT SCENARIO

The single agent scenario demonstrated the effects of the design factors at the most primitive level. The effects of trust, homophily and communication were not seen in this scenario as there were no inter-agent interactions taking place.

1. Civil Security Issue Stance

Figure 4 shows the trend of civil security stance of the single entity I_P_U_S_Sp in the case where RPD is fixed as the only option for decision making method. The 28 plots depict the differences across the 14 different casefiles (7 variants of initial stance and OAB with 2 settings for the effect of scripted actions) and settings for the discount factor. From left to right, the columns correspond to the casefiles with initial stance of 100% inadequate, 99% inadequate, 99% adequate, 50% adequate with 99% PA, 50% adequate with 50% PA, 50% adequate with 99% NA, and 100% adequate. The upper 14 plots are for the cases where the scripted action has a negative effect on the entity, while the lower 14 are for the cases with a positive scripted action effect. The plots on the first and third rows correspond to the discount factor of 0.01, while the second and fourth rows show trends with discount factor set to 0.1. The change in scenario outcome as a result of the scripted action conforms to

[5] Eight output files in comma-delimited value format were generated for each design point, corresponding to 6720 data files in total. Each file contained approximately 4200 to 12600 datapoints, depending on the type and frequency of parameters logged.

expected behavior—the shift in entity perception of civil security issue stance is in the same direction as the effect caused by the periodic scripted action for all test cases.

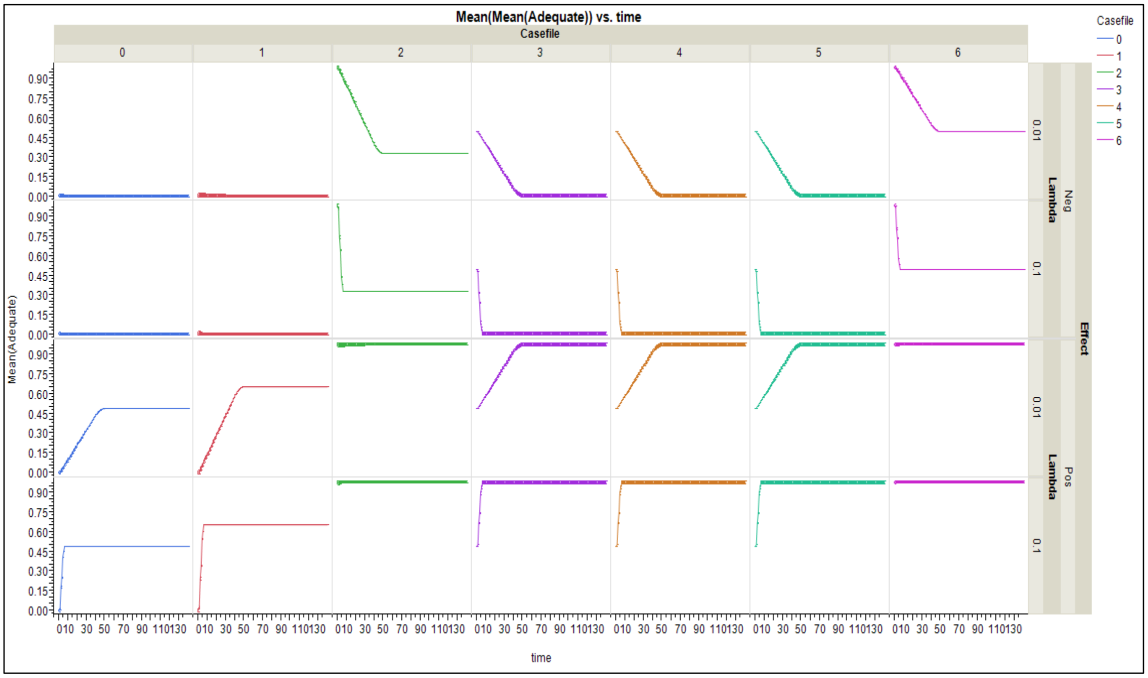

Figure 4. Civil Security Stance over Time - RPD Method.

The variation of both the trend and final state of civil security stance was observed to be unaffected by the decision method adopted by the entity in these test cases. The plots for the settings of EL and BOTH for the decision method were identical to that of the RPD case. This was a clear indication that the decision method was having little or no effect on the final scenario outcome in this set of single agent test cases, which was to be expected, in view of the limited impact that the agent's action selection had in the simple scenario set up.

2. Effect of Initial Stance and OAB

The initial casefiles used for the entity had a significant impact on the scenario outcome. Comparing the cases of 100% inadequate and 99% inadequate, the difference of just 1% resulted in a significant impact on the final level of the issue stance, seen in the bottom left most plots of Figure 4. The same effect was noted in the opposite case, where the initial stance was either 100% adequate or 99% adequate. However, from the 3 casefiles where the population started at 50% level of perceived civil security adequacy, it was noted that the initial OAB towards CF did not cause any change in the final outcome of the scenario. These observations point to the importance of the initial data development process in the CG Model, which constructs casefiles and agent prototypes used in any scenario. The effect of initial stance is further studied in the subsequent test scenarios.

3. Effect of Discount Factor and Size of Dataset

A highly notable observation from the single agent dataset was the significant effect of the discount factor setting on the rate of change of issue stance. Comparing across all test cases with a reinforcement learning discount factor of 0.01, the simulation time required for the issue stance to reach its final steady state was between 3 to 6 days. However, with the discount factor set at 0.1, the time taken ranged from 36 to 49 days. Figure 5 shows the distribution of time taken to reach steady state for replications of the test cases based on an initial stance of 50% adequate, with 50% of the population being positive-active towards CF. The final value of the issue stance was unaffected by the different settings of discount factor. However, it was noted that the issue stance at steady state for the case was affected by the size of dataset used (i.e., the number of respondents on which the casefiles were based). Figure 6 shows the combined effect of the discount factor and number of respondents across the 30 replications of the design point.

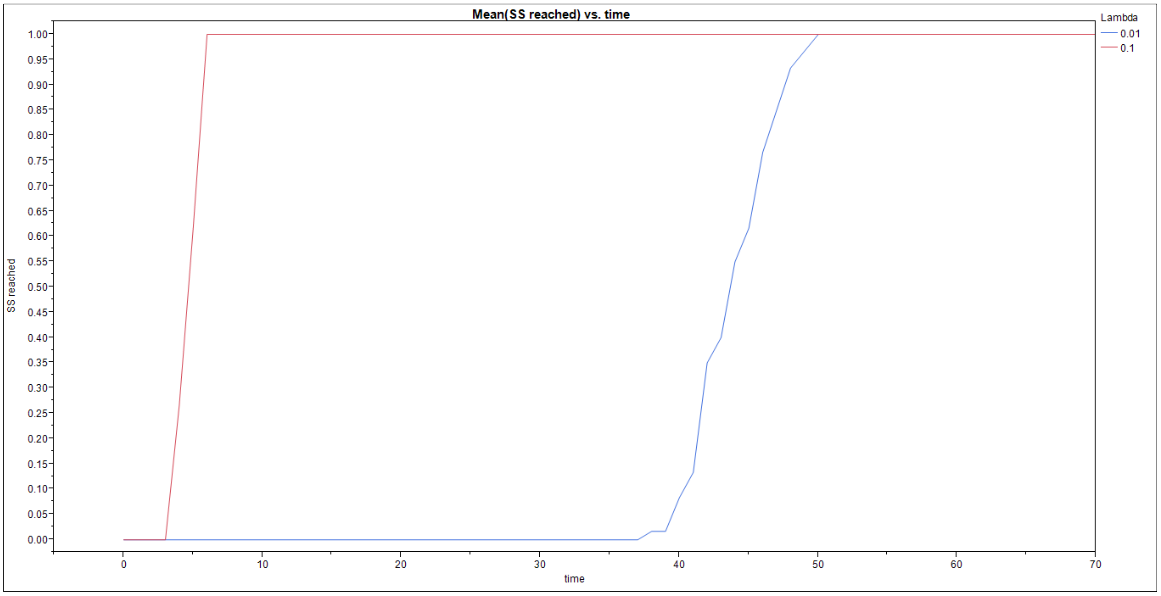

Figure 5. Time Taken to Reach Steady State Outcome in Issue Stance for Different Discount Factor Settings.

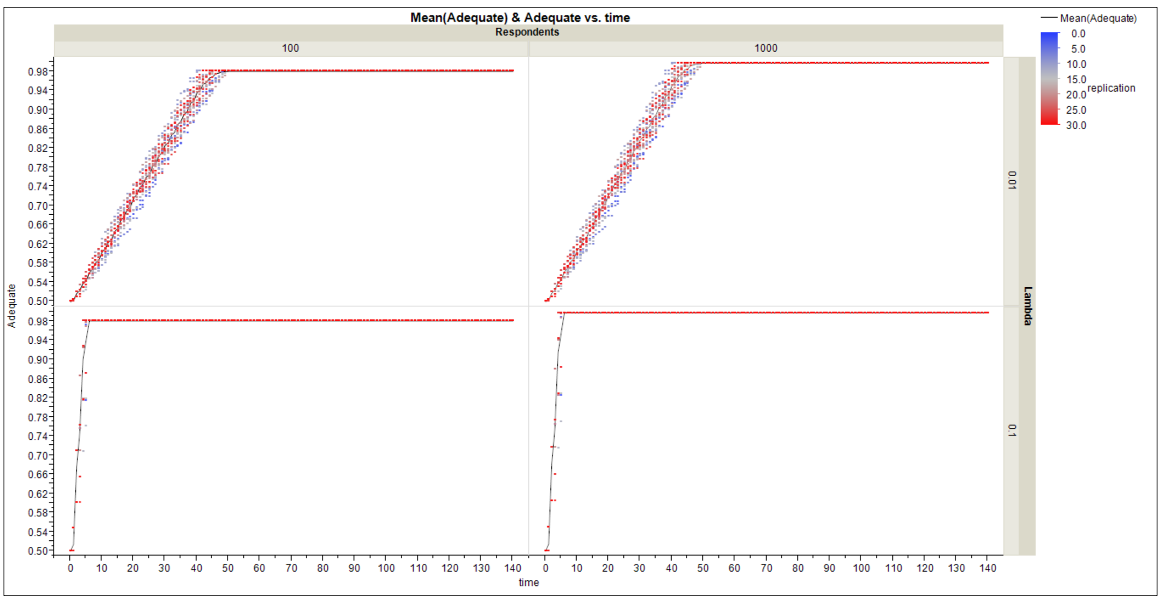

Figure 6. Effect of Discount Factor and Number of Respondents on Civil Security Issue Stance.

E. RESULTS – TWO-AGENT SCENARIO

The results of the two-agent scenario were generally in line with the key observations made from the single agent cases. The data analysis and post processing focused on the design points with the settings of 100 respondents and discount factor of 0.01. This was in consideration of the fact that the cases for 1000 respondents was largely similar to those for 100 respondents, and that the discount factor of 0.1 resulted in behavior (and corresponding scenario outcomes) that shifted too rapidly.

1. Civil Security Issue Stance

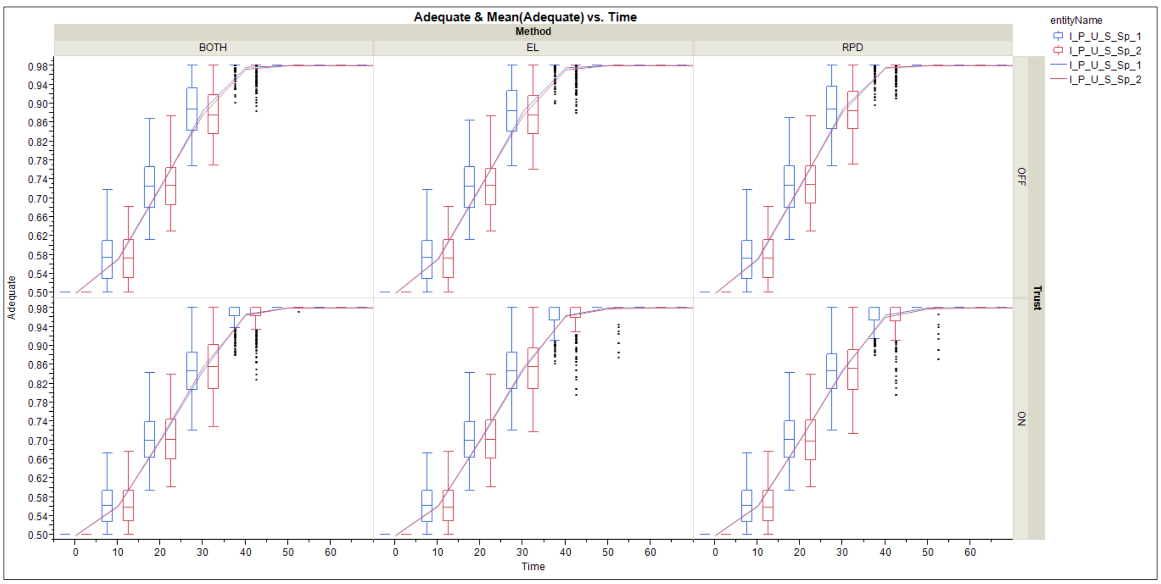

Figure 7. Civil Security Issue Stance for 2-Agent Scenarios.

Figure 7 shows the trend of civil security issue stance over time, for the cases with initial stance at 50% adequacy and positive effect of scripted actions. The stance of both entities remained fairly close to each other throughout the scenario run time, with variations in mean of less than 2% at any point in time. Significant spread was noted across the replications in all six test configurations for the interval in which the stances were shifting from their initial to final states, with a range of up to 22% within each discretized time block of 10 days. The final outcomes and time to reach steady state were comparable to the earlier single agent test cases, with little variation observed between the different decision methods and effects of trust.

2. Decision Method and Action Selection

The effects of decision-making were studied in detail in the two agent scenarios. Figure 8 is a representative plot of the outcomes of decision-making processes for the 50% initial stance cases, showing the experience levels of the entities over time, across the 30 replications of each design point.[6]

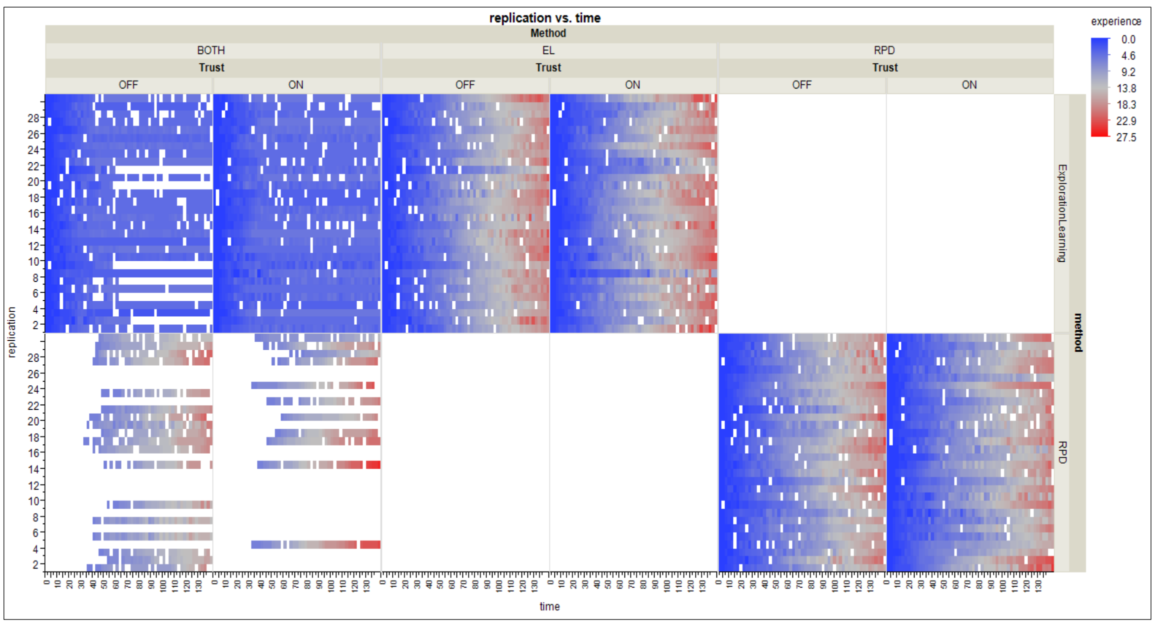

Figure 8. Experience Level Heatmaps over Time

In the design points where the entities could adopt either RPD or EL (heatmaps on left), EL was observed to be the initial choice for decision-making method, as expected. Entity behavior switched to the RPD method for 18 out of 30 replications in the design point where trust was OFF, and 11 out of 30 in the design point where trust was ON. In the cases where EL was maintained throughout the entire duration of the replication, it was observed that the

[6] Blanks within the plots indicate points in time where the event of selecting a particular decision-making method did not occur, and thus no experience level was logged.

experience level of the entities in those runs remained fairly low throughout the scenario. In contrast, with the design points that only allowed EL (plots in center), entity experience continued to rise to significantly higher levels for the majority of replications. Furthermore, the experience that entities attained was comparable to the cases of RPD method only (plots on right).

The observed trend in experience levels of entities using the different decision-making methods highlights a peculiarity of the current implementation of the cognitive architecture. As the RPD method here is essentially a reinforcement learning based technique with a greedy approach, entities that switch to RPD would always select the action that yields the best return. This would suggest that a certain set of actions would consistently not be chosen, if they were associated with the lowest expected utilities, and thus the experience of entities should remain at that value (of the minimum number of times which those actions had been performed). This is clearly not the case in the data observed, as the RPD only cases showed continued rise in experience level, suggesting that other factors are influencing change in behavior or utility of the actions that would otherwise not be used. The EL behavior seen in the plots appear to conform to expectations, with a gradual increase in experience over time, as the entities would be likely to attempt all actions and thus increase the minimum number of times which each has been chosen. These results suggested the need for further study of the decision method selection process and action selection process.

Figure 9 shows the mean expected utilities of the three possible actions pertaining to infrastructure consumption. Agents are able to choose between using their existing service provider ("Use_Current_Provide"), switching to another ("Seek_New"), or decide not to attempt to restock their resources ("Do_Nothing"). The expected utilities for the actions of seeking a new provider or remaining with their existing ones are expected to be similar in this case, as the nodes available to the entities are essentially identical. The trend of expected utilities over time indicate that entity behavior is reasonable in this case—over time, they would continually make the choice of seek out either infrastructure node to resupply themselves, instead of doing nothing. However, it is noteworthy that there is no marked difference for the different decision-making methods or trust settings.

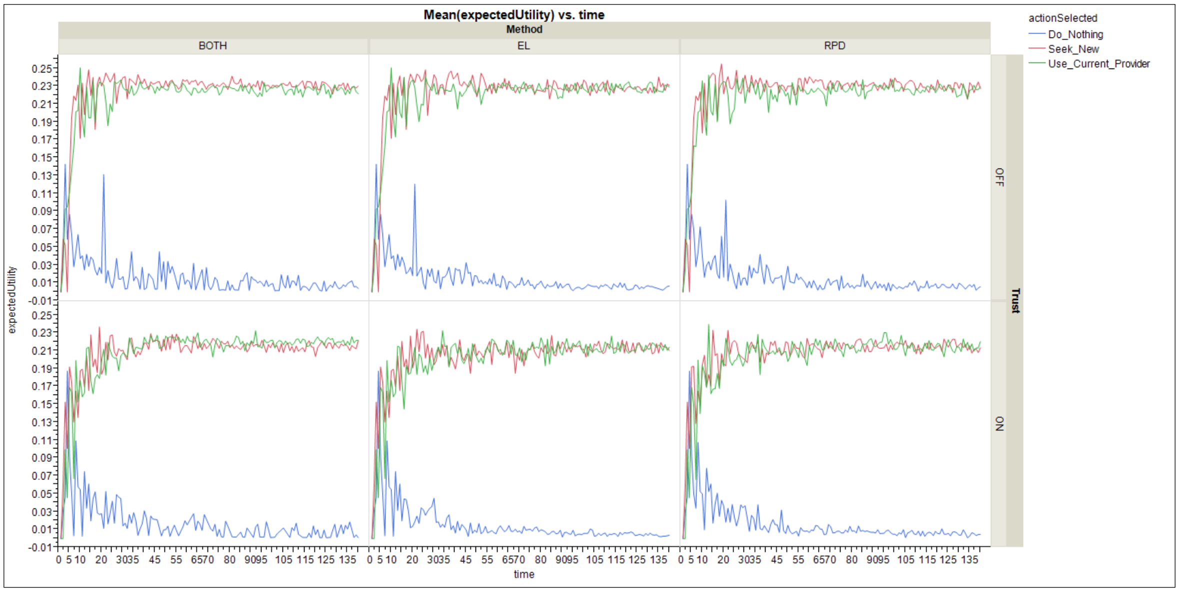

Figure 9. Expected Utility of Infrastructure-related Actions.

3. Homophily and Communications

The homophily link weight between the two entities did not vary with the different decision methods and trust settings. However, the effect of the trust was observed from its effect on communications between the entities. The initial trust level between the entities in these cases was set at 0.5, which rapidly increased to close to the maximum of 1.0 as expected, given the high degree of homophily between them (since they are built on the same prototype). The percentage of communications between the entities that were accepted thus increased over time, from an initial 66% to 87% by the end of the simulation (Figure 10).

		commDecision	
	previousTrust	RECEIVE_ACCEPT	RECEIVE_DONT_ACCEPT
Time	Mean	Row %	Row %
10	0.67295154	66.22%	33.78%
20	0.8845323	81.87%	18.13%
30	0.95944339	87.28%	12.72%
40	0.97715337	87.56%	12.44%
50	0.984732	87.01%	12.99%
60	0.9887895	86.99%	13.01%
70	0.98887569	86.53%	13.47%
80	0.99190848	88.01%	11.99%
90	0.99192274	86.95%	13.05%
100	0.99084663	87.41%	12.59%
110	0.99201641	88.56%	11.44%
120	0.99276436	87.62%	12.38%
130	0.99356156	87.88%	12.12%
140	0.99254373	87.19%	12.81%

Figure 10. Communications Acceptance/Rejection Rate.

F. RESULTS – THREE-AGENT SCENARIO

1. Civil Security Issue Stance

The civil security stance in the 3-agent scenario showed a similar trend over time as that of the 2-agent case (Figure 11).

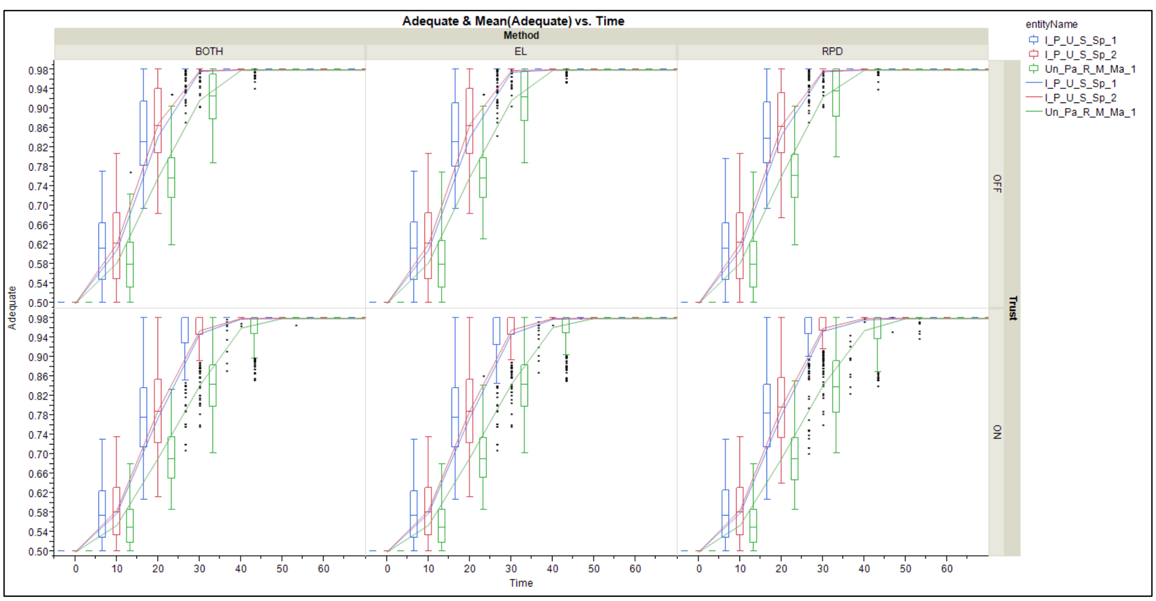

Figure 11. Civil Security Issue Stance for 3-Agent Scenarios.

The new agent, Un_Pa_R_M_Ma_1 demonstrated behavior similar to the original two, but took a longer time to reach its final state in issue stance. The effect of communication was clearly the cause of this behavior—at the 40 day mark, the Un_Pa_R_M_Ma_1 entities in the test cases where the trust module was deactivated had all reached steady state of 98% adequate. In contrast, for the cases with trust on, the mean issue stance in the same time period was 96%, with a 3% standard deviation and range from 87% to 98%. Figure 12 and Table 5 compare the standard deviation of issue stance over time under the effects of trust. The variance is significantly increased for all cases where the trust module is active, but not affected by the decision method used.

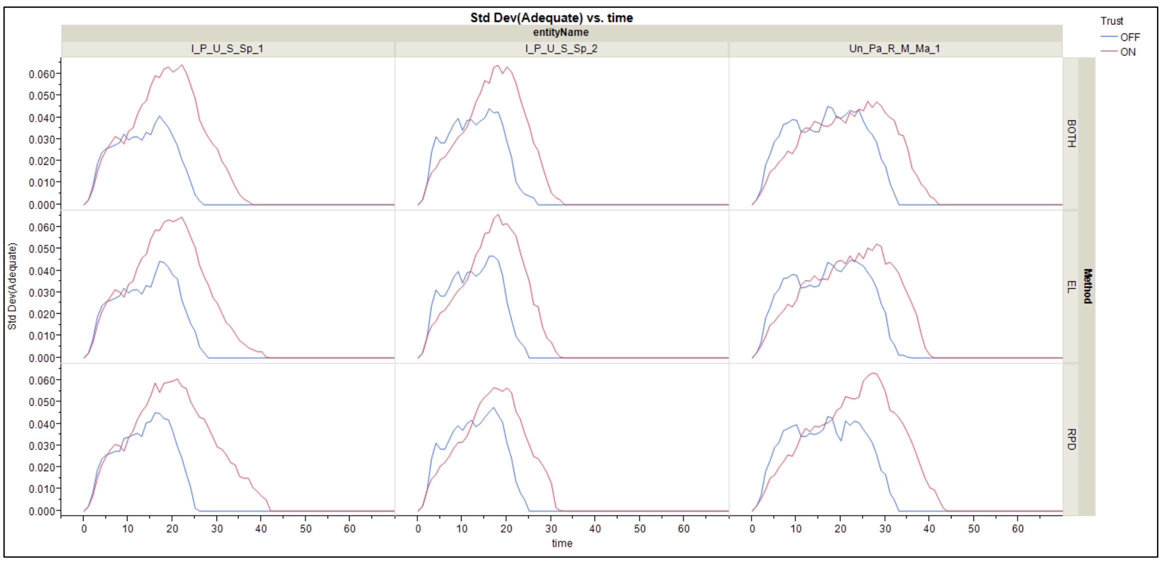

Figure 12. Effect of Trust on Deviation in Issue Stance.

Entity	Trust	Max. Range	Peak Std Dev.	Max.Time to Steady State
I_P_U_S_Sp_1	ON	30.4% (Day 19)	6.5% (Day 22)	Day 43
	OFF	18.4% (Day 15)	4.5% (Day 16)	Day 28
I_P_U_S_Sp_2	ON	27.2% (Day 17)	6.6% (Day 18)	Day 32
	OFF	20.8% (Day 15)	4.8% (Day 17)	Day 27
Un_Pa_R_M_Ma_1	ON	21.5% (Day 26)	6.4% (Day 27)	Day 44
	OFF	18.9% (Day 10)	4.5% (Day 17)	Day 34

Table 5. Effect of Trust on Range and Deviation of Issue Stance.

2. Decision Method and Action Selection

The experience levels of the three entities were comparable throughout the progress of the scenario, and the results showed behavior similar to the 2-agent cases. Additionally, as seen in Figure 13, the trend of experience gain by entities in RPD or EL only modes was distinctly different from the cases where both decision methods were admissible. As before, the expected behavior in EL mode matched the experience trend observed, but that of RPD mode did not. These findings reinforce the notion that the implementation of RPD in the CG Model is in essence a reinforcement learning type approach, but also point out that the process of choosing between EL and RPD alters the behavior of the entities such that the outcome differs from a pure EL or pure RPD scenario.

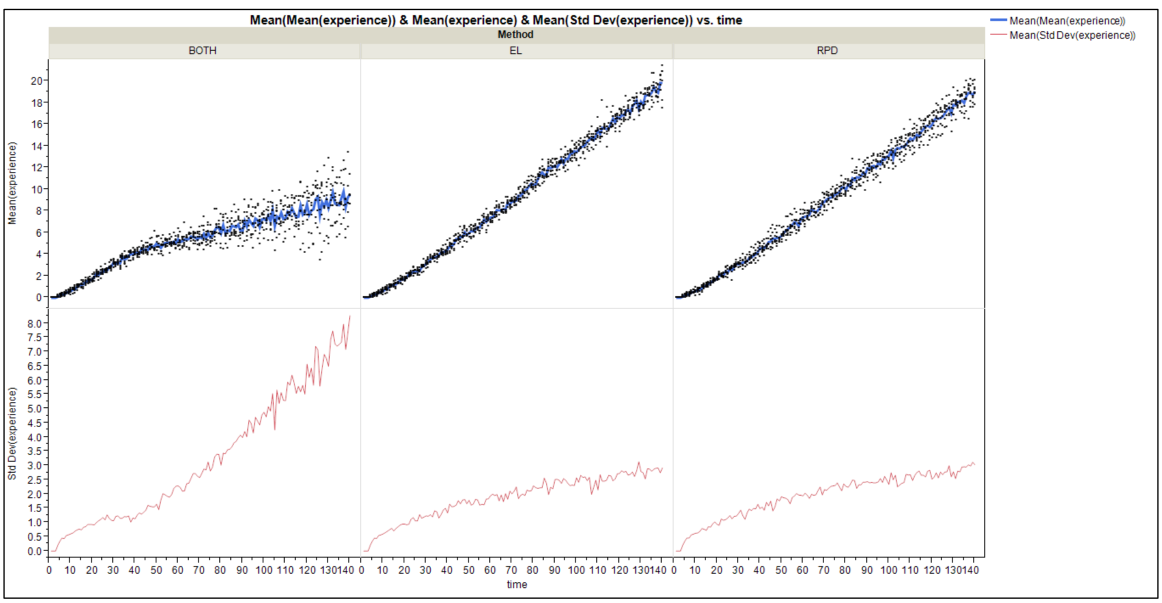

Figure 13. Entity Experience over Time.

3. Homophily and Communications

The degree of homophily was expected to differ between the I_P_U_S_Sp entities and the single Un_Pa_R_M_Ma entity. The earlier data indicating the slower response of the Un_Pa_R_M_Ma in terms of civil security issue stance pointed to the possibility that it was not receiving communications as readily due to its lower homophily link weigh with the other entities. The data shown in Figure 14 provides some evidence of this behavior, indicating that communications between I_P_U_S_Sp and Un_Pa_R_M_Ma averaged at an acceptance rate of 85.4%. In comparison, the communications between the I_P_U_S_Sp entities was accepted 86.1% of the time. More significantly, the volume of communications between I_P_U_S_Sp entites averaged 1.21 times a day, against 0.94 times a day for Un_Pa_R_M_Ma_1 to either of the other two entities. This indicated that the effect of homophily (determining the entities' desired to communicate with each other) was far more significant compared to trust (which determined acceptance of communications received). Comparison of the homophily link weights and trust levels between entities did not yield any other new findings.

		commDecision			
		RECEIVE_ACCEPT		RECEIVE_DONT_ACCEPT	
sender	receiver	N	Row %	N	Row %
I_P_U_S_Sp_1	I_P_U_S_Sp_2	15261	86.55%	2371	13.45%
	Un_Pa_R_M_Ma_1	11323	85.57%	1909	14.43%
I_P_U_S_Sp_2	I_P_U_S_Sp_1	15261	85.67%	2553	14.33%
	Un_Pa_R_M_Ma_1	10899	85.26%	1885	14.74%
Un_Pa_R_M_Ma_1	I_P_U_S_Sp_1	10558	84.61%	1921	15.39%
	I_P_U_S_Sp_2	14608	85.98%	2382	14.02%

Figure 14. Communications Acceptance/Rejection Rates Between Entities in 3-Agent Scenario.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. FURTHER TESTING AND EVALUATION

A. DESIGN PARAMETERS

The results and analysis of the initial set of design points suggested that the effects of decision method and trust were being overshadowed by other design factors in the model. The next phase of the testing and evaluation was thus developed to maximize the possible effects from these components of the cognitive architecture. In addition, factors that were found to be less significant or less relevant to test purposes were removed. The discount factor was fixed at 0.01, and only the casefiles based on 100 respondents were used.

The initial issue stance and OAB of entities was seen to have significant influence on the behavior and effect on scenario outcome. Several levels were tested, of which four were chosen for final set of design points. Most importantly, the periodic scripted action effect was removed and replaced with single action, as described in test scenario description in the next section. Table 6 shows the 24 design points that were used for the final run.

Design Point	Decision Method	Trust	Initial Stance	Design Point	Decision Method	Trust	Initial Stance
951	RPD	ON	99% Adequate	963	RPD	ON	55% Adequate
952		OFF		964		OFF	
953	EL	ON		965	EL	ON	
954		OFF		966		OFF	
955	BOTH	ON		967	BOTH	ON	
956		OFF		968		OFF	
957	RPD	ON	75% Adequate	969	RPD	ON	50% Adequate
958		OFF		970		OFF	
959	EL	ON		971	EL	ON	
960		OFF		972		OFF	
961	BOTH	ON		973	BOTH	ON	
962		OFF		974		OFF	

Table 6. Design Points for Final Run.

B. TEST SCENARIO

Six agents were utilized for the final round of testing. These comprised three I_P_U_S_Sp and three Un_Pa_R_M_Ma entitites. The scenario was also expanded geographically – the two infrastructure nodes were placed at a distance of about 10 hex-grids apart, and the agents were distributed around them as shown in Figure 15. Each grid represents an area of approximately 1-mile radius.

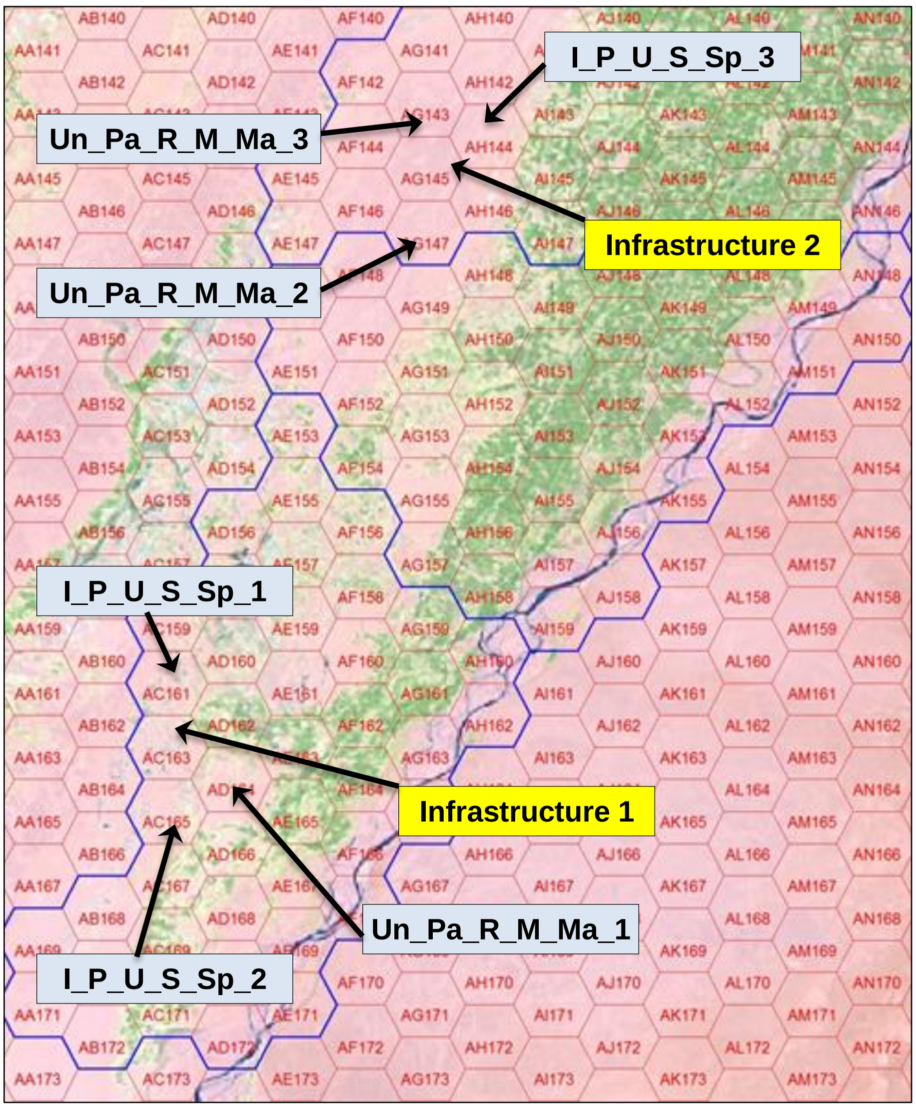

Figure 15. Map of Area of Operations (From Yamauchi, 2012).

With this set up, the effects of geographical location, communications between entities regarding infrastructure, and success rates of visiting the nodes will come into play. The effect of infrastructure visits was adjusted to have variable impact on entity stance—if an agent succeeds in restocking when he visits a node, there would be a 75% likelihood for a positive effect on stance, and a 25% otherwise. However, this is only one of the factors determining any overall change in stance, because the influence of other parameters also contributes to overall behavior choices and net change in issue stance.

The periodic scripted action used previously was replaced by a single action that occurred at a fixed time. The scenario was initialized with one of two infrastructure nodes *inoperable*, and the other at a *minimal* state (Table 7 provides the definition of infrastructure operation states). At day 90 of the scenario, the scripted action for CF to improve the inoperable infrastructure node takes place, restoring its state to *normal*. The operation state of the other node remains *minimal*. This setup causes entities to fail if they attempt to restock consumables from the first node prior to day 90, and to periodically fail when they attempt to restock from the second node throughout the scenario (essentially, only 1 of 7 attempts would succeed).

State	openTime	closeTime	numberServers	queueCapacity
Normal	360	0	1	10
Reduced	2	5	1	10
Minimal	1	6	1	10
Inoperable	-	-	-	-

Table 7. Definitions for Infrastructure Operation States.[7]

[7] Several configurations for the initial state and state after scripted repair action were tested to develop this set of parameters and scenario settings, such as varying the queue capacity, transfer rates and resource capacity of the nodes. These settings mean that the node at *minimal* state will be available for 1 out of every 7 days. Entities attempting to restock on the days that it is closed will experience a failure in the action. Those visiting on the day it is open will most likely receive their requested resource, as the server and queue capacity is sufficient to provide for all entities in the scenario (unless balking or reneging occurs due to other entities being in the queue ahead of it). The *inoperable* state always fails to provide resource to the visiting entity.

Thus, the expected behavior is for entities to initially experience a decline in stance, due to the inability to receive the requested resource. Also, the choice of actions would favor Node 2 over Node 1. After the action of infrastructure improvement, Node 1 becomes more viable of the two, and agents who maintain exploratory behavior are expected to realize this, possibly communicate with other entities, and thereby cause action choices to shift in favor of Node 1. The effect on stance is expected to be favorable, since the entities would then experience a high success rate, and thus the overall scenario outcome should show an improvement of issue stance over time.

The scenario length for this set of tests was increased to 360 days, allowing for trends and outcomes to stabilize and possibly reach their steady state levels. Thirty replications were run for each design point, using the same seeds as before.

C. OUTPUTS

Additional dataloggers were used for this set of tests (Table 8), including new code that was added to the ongoing revisions of the CG Model. In particular, the BehaviorEffects-Datalogger was added to track all occurences of entities visiting either infrastructure, and capture their success/failures as well as the resultant effect on their issue stance.

Parameter	Datalogger(s) Used	Description
Infrastructure Visits	BehaviorEffects-DataLogger	Record of infrastructure visits on both nodes, outcome (succeed / fail), and effect on civil security issue stance (increase / decrease / unaffected).
Other Parameters	Location-DataLogger State-DataLogger Behavior-DataLogger Action-DataLogger	Additional parameters were recorded for cross-referencing and checking purposes. These were the locations of entities (to check entity movement around the area), state of infrastructure nodes, behavior choices of entities and occurrence of scripted actions.

Table 8. Description of Additional Key Parameters Measured.

D. RESULTS

1. Civil Security Issue Stance

The effect of initial population stance on the scenario outcome is clearly visible in Figure16. As expected, initial trend in civil security is negatively-sloped, given that the infrastructure in the scenario is unable to provide consumables for the entities most of the time. The introduction of the scripted event at Day 90 triggered the change in behavior, seen as either a reduction of the decline in issue stance, or a change in the direction of the trend.

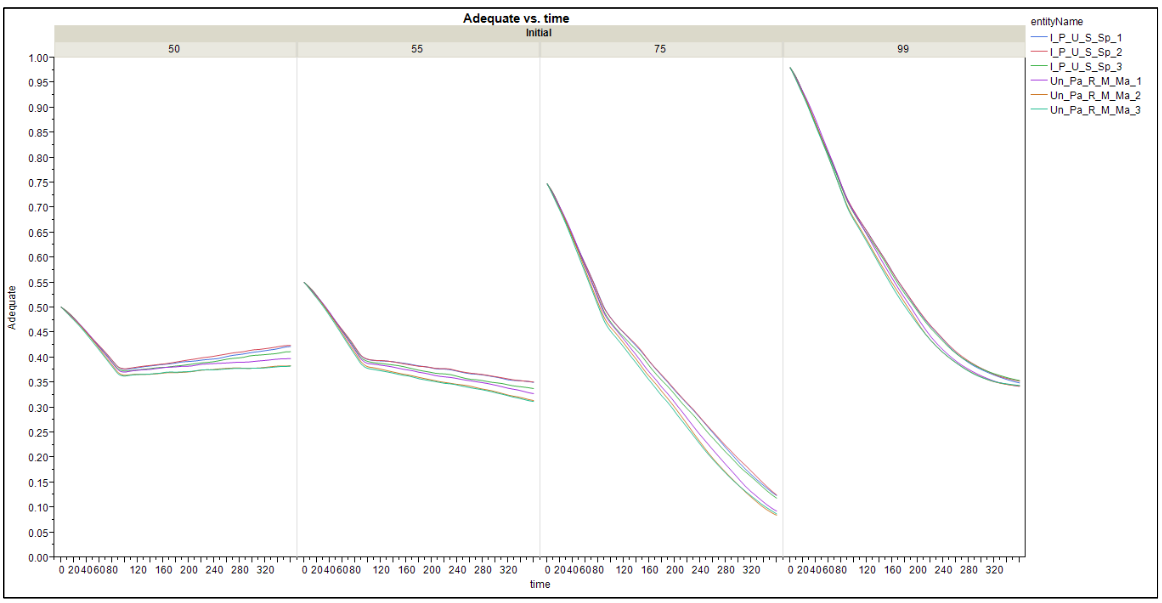

Figure 16. Civil Security Issue Stance for Different Initial Stance Levels.

In the CG Model, the initial issue stance determines the base effect from which the change caused by future actions are calculated. This implementation is responsible for the phenomena seen above, whereby the cases with a very high initial issue stance appears to be least affected by improvements brought about after the scripted action occurs. Further discussion of these effects is presented with the results of entity behavior and action selection in the next section.

Considering the case of 50% initial stance as an example (Figure 17), the decision method alone did not demonstrate significant effect on scenario initially. The trend of civil security issue stance over time for all entities followed a tightly bound range up till the point when the scripted action occurred. However, the effect of trust reduced the rate of change of entities' issue stances, resulting in a highly percentage of adequacy at the time the scripted action occurs. After day 90, the increase in choices available to the entities generated sufficient variation in the action-selection process to cause some degree of spread in the outcome at the end of the scenario as compared to the earlier simple scenarios. Figure18 and Table 9 provide the breakdown of the civil security issue stance at the conclusion of the test scenario (day 360) for the 6 configurations of decision methods and trust. The results indicate that the overall scenario outcome is better (i.e., a higher percentage of the population feel that civil security is adequate) when the entities used both RPD and EL methods, compared to only one particular decision method.

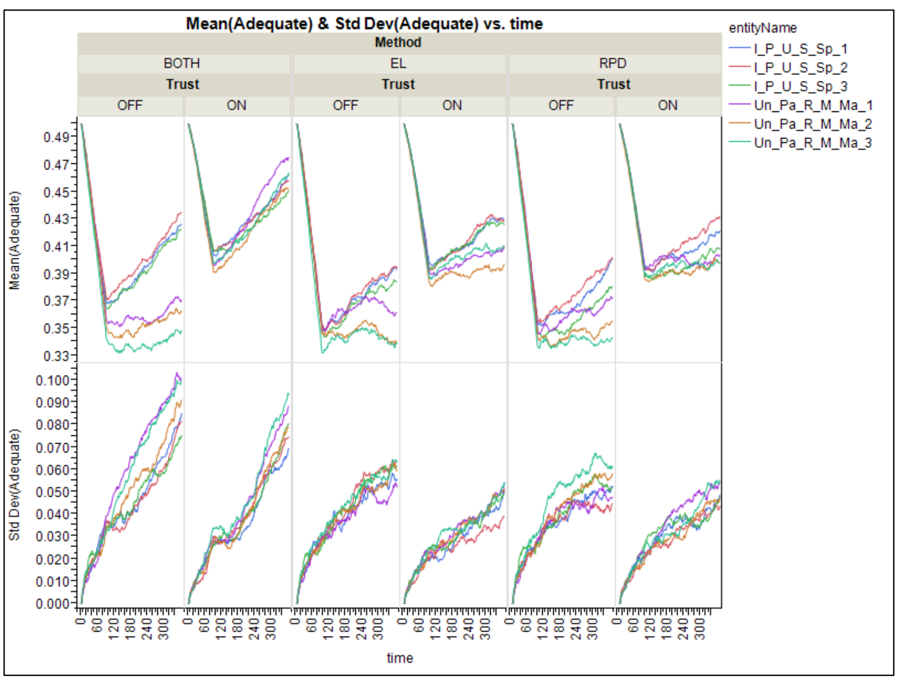

Figure 17. Civil Security Issue Stance for Initial 50% Adequate.

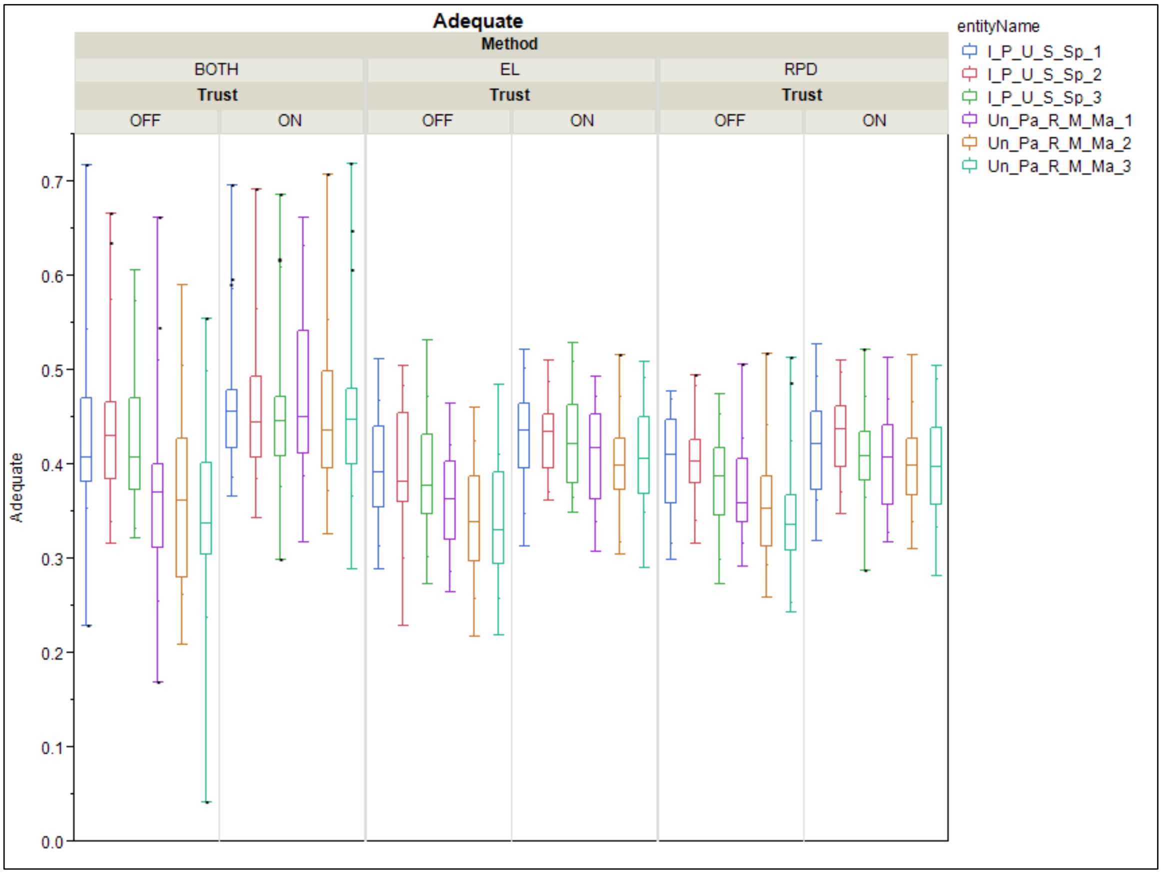

Figure 18. Distribution of Outcomes - Civil Security Stance at Day 360.

Configuration		Mean Stance (% Adequate)	Standard Deviation	95% Confidence Interval	
Method	**Trust**			**Lower Bound**	**Upper Bound**
BOTH	OFF	39.4%	9.5%	38.0%	40.8%
	ON	46.1%	8.1%	44.9%	47.3%
EL	OFF	36.9%	6.3%	36.0%	37.8%
	ON	41.7%	5.1%	41.0%	42.4%
RPD	OFF	37.6%	5.7%	36.8%	38.4%
	ON	41.0%	5.1%	40.3%	41.7%

Table 9. 95% Confidence Interval Levels of Civil Security Stance at Day 360 (Combined Mean across all Entities in Scenario).

2. Decision Method and Action Selection

The infrastructure-related choices made by entities in the final scenario provided further insight to their behavior and the effects of the decision methods. The actions selected and resultant effects are summarized in Figure 19, which includes the data from all 24 design points.

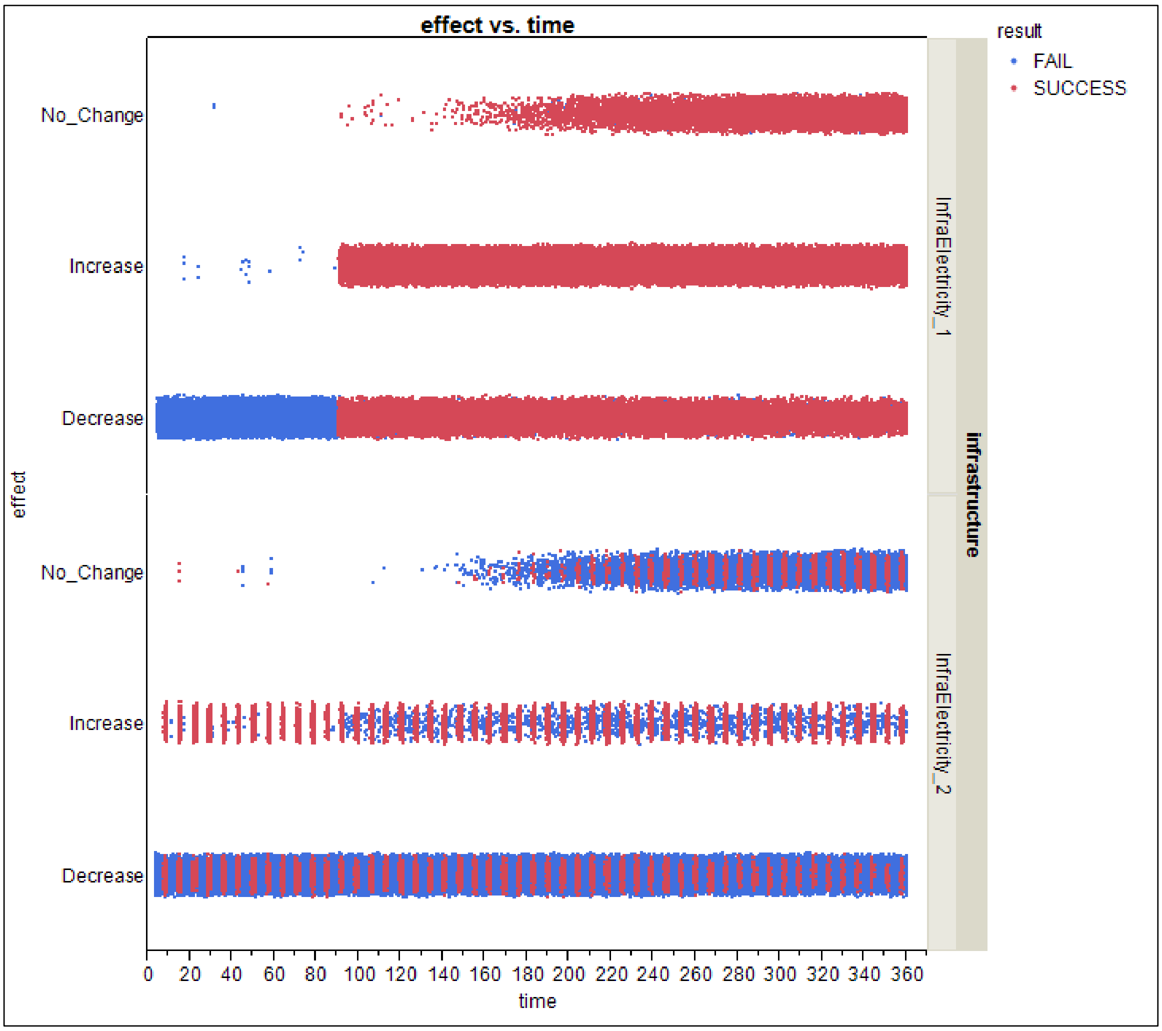

Figure 19. Infrastructure Node Visitation Outcomes and Effects.

The behavior of the entities provides a key insight that the outcome of an entity's visit to a node can generate both positive and negative effects on its issue stance, regardless success or failure to obtain the resource requested. In particular, during the second half of scenario run time, there is a significant increase in instances of actions that do not cause any change to stance. The visitation rates of the two infrastructure nodes (Figure 20) provide a tell-tale sign that entity behavior is not ideal in the model / scenario—despite an total failure rate of 86.2% experienced with infrastructure node 2, entity behavior does not change to avoid it, as would be expected for a reinforced learner.

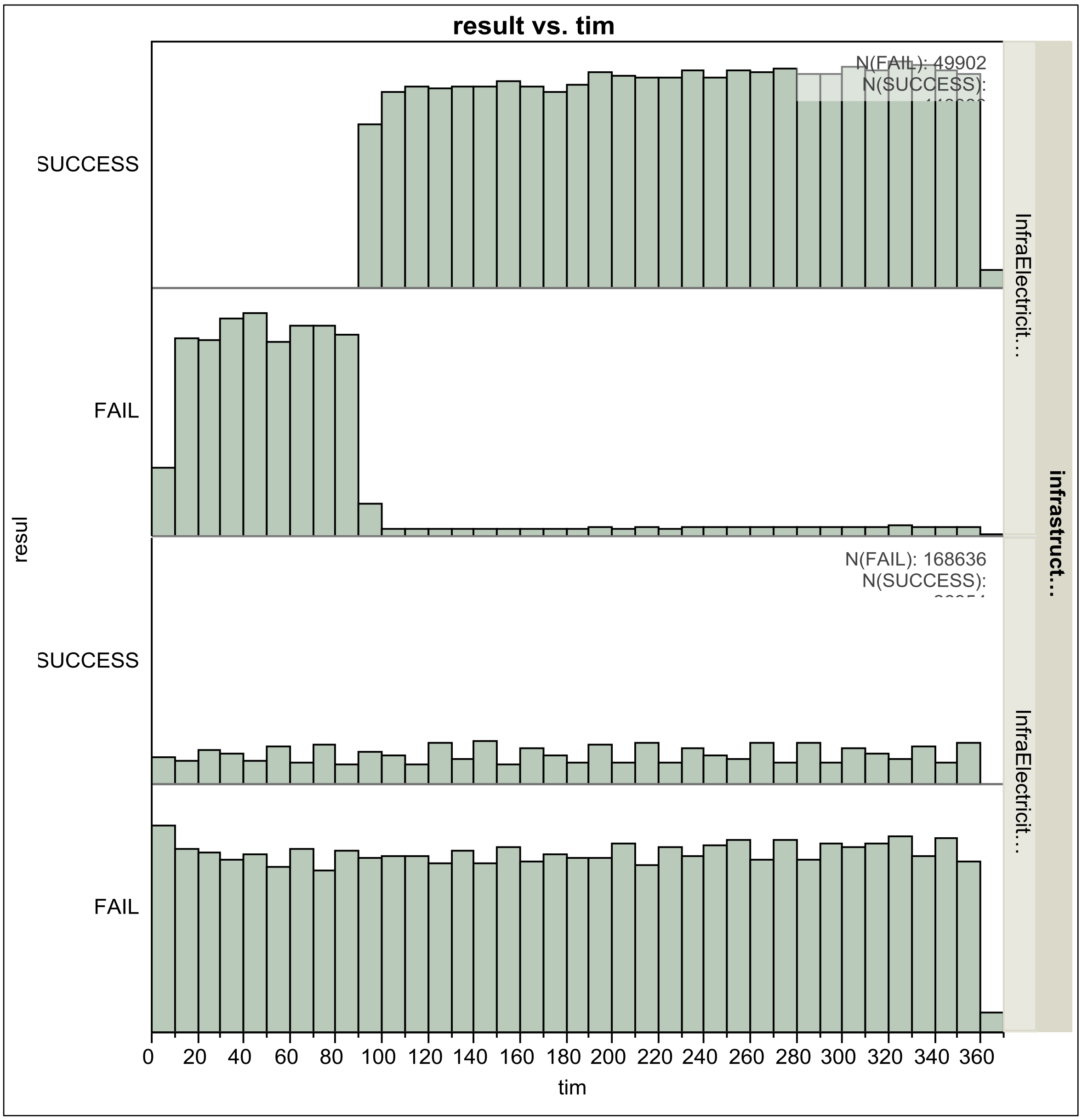

Figure 20. Infrastructure Node Visitation Rates and Outcomes.

Data from the action-selection process was used to investigate the cause of such agent behavior. Figure 21 plots the expected utilities of the three possible infrastructure-related actions on a logarithmic scale for all 24 design points in the scenario. The increase in expected utility of seeking a new provider corresponds to the occurrence of the scripted action at day 90; however, the action of remaining with an entity's existing provider also increases in value over time. This trend results in agent behavior that does not focus on either choice.

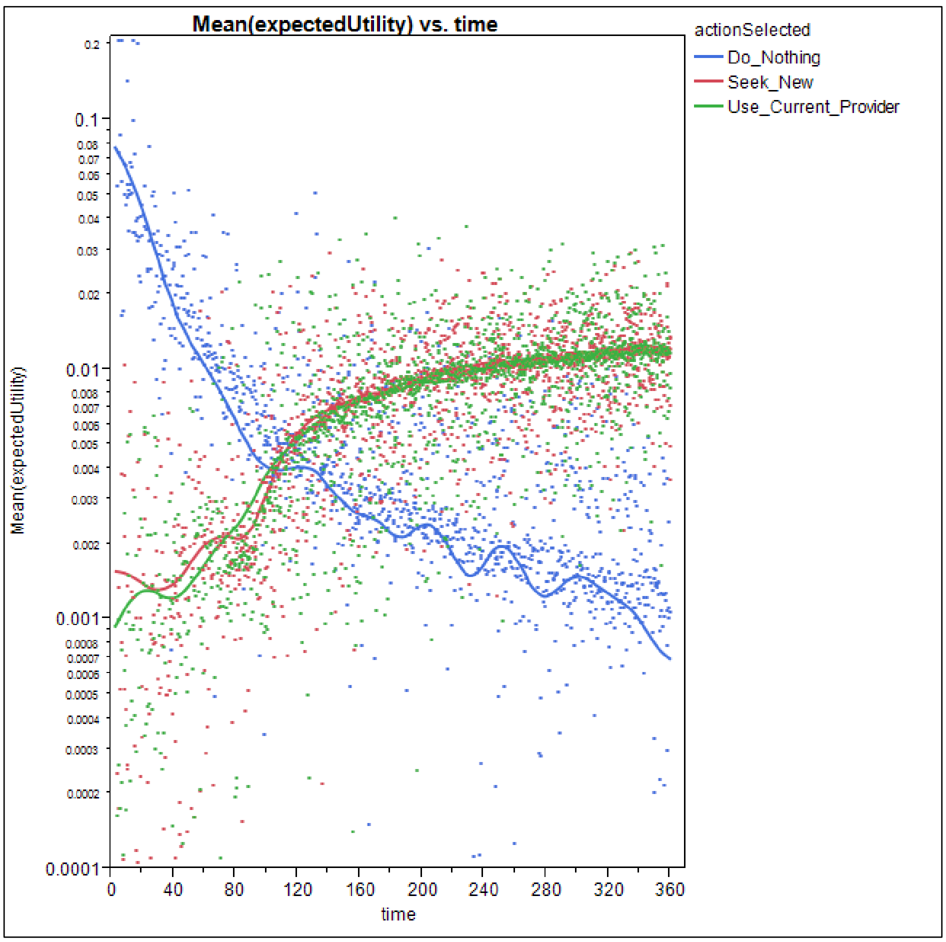

Figure 21. Expected Utility of Infrastructure-related Actions in 6-Agent Scenario.

Further analysis of the source code and consultation with the programmer (H. Yamauchi, personal communication, July 2012) revealed that the existing algorithm for allocation of rewards to the actions does not account for the state of the entity, which explained the behavior observed in the infrastructure-related action selection process. Entities that visited a node and received an unfavorable outcome would have a higher probability of choosing to seek a new provider on their next action selection. However, upon switching to the better node, the expected utility for seeking a new node would be higher than the action of staying with that new provider. The resultant behavior would cause the agent to switch back and forth between nodes, seemingly with no regard to the outcomes from the infrastructure visits.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSION

The CG Model utilizes a highly complex cognitive architecture module in order to accurately and realistically depict the behavior of civilian populations in an IW environment. The critical process of entity decision making is based on well-accepted social science theories that provide a sound framework for the artificial intelligence of entities. The decision methods and trust module used in the CG Model were found to perform adequately, despite some deviations from expected behavior that were attributed to limitations in the implementation of these conceptual models.

A. EFFECTS OF DECISION METHOD

The process of decision method selection in the CG Model utilizes a reinforcement learning algorithm in two ways—as an exploratory approach, to allow entities to try out possible actions and build up their knowledge of expected utilities; and as a greedy approach, to simulate a RPD model of decision making. The test scenarios showed that the EL approach was adequate in generating agent behavior which performed as expected. The RPD approach generated similar scenario outcomes to the EL mode, in terms of overall trend and end state of civil security issue stance, behavior actions and interactions between entities. The combination of both methods, as implemented in the existing CG Model, generated scenario outcomes over a far larger range of possibilities, with close to twice as much variation as compared to either RPD or EL alone. However, the mean outcome was shown to be fairly similar across the design points tested. The effect of other parameters, in particular the initial stance of the entities, was far more significant in influencing the overall stance at the end of the scenario.

The significant increase in variance generated when both RPD and EL methods are used suggests that this implementation would be useful for the purpose of exploring potential outcomes for any given set of inputs, as it would cover a larger sample space.. However, continued development to independently refine the RPD method would also be important to allow the model to better capture the effects of ‘expert’ entities (vis-à-vis a novice that would require several rounds of exploratory behavior to attain the same experience). Also, the existing cognitive architecture has limitations in associating utilities to state-action pairs instead of actions alone, which resulted in behavior that deviated from expectations, but still allowed entities to make choices and influence the outcome of the scenarios in a coherent manner.

B. EFFECTS OF TRUST

The inclusion of the trust module in the CG Model was shown to have a strong influence on the rate of change in issue stance of entities. This collaborates with the findings in Pollock’s (2011) implementation; however, the outcomes of the test scenarios were shown to converge towards the same steady state regardless of the trust setting. The trust module thus serves as a buffer that delays the impact of actions in the area of operations, as its current form (as used in the test scenarios) only act to reject information. However, there is potential for it to influence scenario outcome, depending on the time frame allocated, and the frequency of actions occurring in the scenario.

C. OTHER FACTORS

The initial test scenarios demonstrated the strong impact that input parameters for a CG Model scenario can have. In line with the findings of earlier studies (Papadopoulos, 2010; Pollock, 2011), careful selection of these factors is crucial in order to build a realistic scenario that matches user requirements and expectations of agent behavior. The test cases showed, in particular, that the initial stance of the population was extremely significant.

D. TRACEABILITY OF ENTITY BEHAVIOR

The complexity of interactions in the CG Model makes tracing of entity behavior rather challenging. The process adopted in this study demonstrated the need to explore effects of different components of the CG at multiple levels, ranging from the isolation of single factors to larger scenarios with multiple parameters being evaluated. The dataloggers built into the existing CG Model served as valuable tool for recording the immense amount of data generated in each replication and design point.

The experimentation done in this thesis has assisted the ongoing development of the CG Model. Several revisions of the code were made to adjust settings and rectify minor anomalies in the entity behaviors. The creation of new dataloggers by TRAC-MTRY programmers would also provide for future testing and evaluation efforts, and improve the traceability of entity behavior.

E. FUTURE WORK AND RECOMMENDATIONS

The analysis of the effects of decision methods in the CG Model revealed a few aspects of the cognitive architecture that could be improved. The greedy reinforcement learning approach used for the RPD method and the limitation on state-action pair association in the EL method are two key areas that could be investigated for future developments.

In terms of analysis and testing of the cognitive architecture, several areas have been identified that could benefit from further study:

1. The test scenarios used in this study utilized only two entity prototypes, which posed a constraint on the extent of differences in homophily and possible interactions between them. Expansion of the scenario to include more agent types would serve to test the effect of homophily and communications to a greater extent.

2. The EL method is applicable to a wide range of actions that entities could undertake in the CG Model. The testing of infrastructure-related actions in this study was limited by the lack of accounting for entities' existing states (current resource provider). Testing of the EL method in other contexts, in

particular for scenarios or actions that are less/not dependent on state would serve to build up further understanding of the action selection process in the CG Model.

3. The current implementation of trust in the CG Model acts to restrict information flow to an entity. An opposite effect could be modeled such that an entity receiving percepts from a highly trusted counterpart would be influenced to a greater extent than normal. This would allow shifts in scenario outcomes in either direction as a result of trust, instead of the single-direction “buffering” effect that was observed in this study. However, such an implementation would increase the complexity of the CG Model even further.

This study has shown that the decision methods and trust module in the cognitive architecture are significant components in the CG Model. However, their effects are not always visible in terms of measurable outcomes such as issue stance of entities and overall trends in agent behavior. The test scenarios involved simplistic settings and did not exhibit any degradation of performance (e.g., computation / simulation time). However, with full-scale wargaming scenarios, the removal or deactivation of some components may become an acceptable tradeoff.

LIST OF REFERENCES

Ajzen, I. (1985). From intentions to actions: A theory of planned behavior. In J. Kuhl & J.Beckmann (Eds.), *Action-control: From cognition to behavior* (pp. 11–39). Heidelberg, Germany: Springer.

Ajzen, I. (1991). The theory of planned behavior. *Organizational Behavior and Human Decision Processes*, *50*, 179–211.

Alt, J. K. (2010). Agent frameworks for discrete event social simulation. *Proceedings of the 19th Conference on Behavior Representation in Modeling and Simulation*. Charleston, SC.

Alt, J. K., Jackson, L. A., Hudak, D., & Lieberman, S. (2009). The cultural geography model: Evaluating the impact of tactical operational outcomes on a civilian population in an irregular warfare environment. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, *6*(185), 185–199.

Baez, F., Alt, J., Brown, R., Lieberman, S., Ehlschlaeger, C., Zwicky, W. R., … Yamauchi, H. (2010). *Cultural Geography (CG) Model Enhancements for Irregular Warfare (IW) Analysis*. Monterey, CA: TRAC-MTRY.

Buss, A. (2011). *Discrete Event Simulation Modeling* (Technical report). MOVES Institute, Naval Postgraduate School.

Department of Defense. (2008). *Irregular warfare* (DoD Directive 3000.07). Retrieved from http://www.dtic.mil/whs/directives/corres/pdf/300007p.pdf

Ferris, T. P. (2008). *Modeling methodologies for representing urban cultural geography in stability operations* (Master's thesis). Naval Postgraduate School.

Fisher, W. R. (1984). Narration as a human communication paradigm: The case of public moral argument. *Communication Monographs*, *51*(1), 1–22.

Hudak, D. & Baez, F. (n.d.). *Cultural geography modeling and analysis in helmand province*. Technical Report. Monterey: TRAC-MTRY.

Joint Chiefs of Staff. (1995). *Joint doctrine for military operations other than war* (Joint Publication 3–07). Washington, D.C.: Joint Staff Pentagon

Klein, G. (1999). *Sources of power: How people make decisions*. Cambridge, MA: MIT Press.

Klein, G. A., Calderwood, R. & Clinton-Cirocco, A. (1986). Rapid decision making on the fire ground. *Proceedings of the Human Factors Society 30th Annual Meeting*.

Klein, G. A. (1998). A recognition-primed decision model of rapid decision making. In Klein, G.A., *Recognition-primed decisions* (pp. 138–147). Retrieved from Defense Technical Information Center OAI-PMH Repository.

McPherson, M., Smith-Lovin, L. & Cook, J. M. (2001). Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, *27*, 415–444.

Modeling & Simulation Coordination Office [MSCO]. (n.d.). Home Page of the Department of Defense Modeling & Simulation Coordination Office. Retrieved from http://www.msco.mil/descMSCO.html

Ng, E. H. (2012). Keynote address by Minister for Defence Dr Ng Eng Hen at the *2012 Singapore Armed Forces Learning Symposium*. Singapore.

Ozcan, O. (2011). *Balancing exploration and exploitation in agent learning*. (Master's thesis). Monterey, California: MOVES Institute, Naval Postgraduate School.

Ozcan, O., Alt, J., & Darken, C. J. (2011). Balancing exploration and exploitation in reinforcement learning. *Proceedings of the Twenty-Fourth International Florida Artificial Intelligence Research Society Conference*. FL.

Papadopoulos, S. (2010). *Reinforcement learning: A new approach for the cultural geography model* (Master's thesis). Monterey, California: MOVES Institute, Naval Postgraduate School.

Perkins, T. K., Pearman, G. M., Baez, F. (n.d.). *Representing civilian population interaction with infrastructure and essential services in stability operations*. Monterey, CA: TRAC-MTRY.

Pew, R. W., & Mavor, A. S. (Eds.). (1998). *Modeling human and organizational behavior: Application to military simulations*. Washington, D.C.: National Academy Press.

Pollock, S. (2011). *Representing trust in cognitive social simulations* (Master's thesis). Monterey, California: MOVES Institute, Naval Postgraduate School.

Proctor, R. W. & Zandt, T. V. (2008). *Human factors in simple and complex systems* (2nd ed.). Boca Raton, FL: CRC Press.

Russell, S. & Norvig, P. (2010). *Artificial intelligence: A modern approach* (3rd ed.). Upper Saddle River, NJ: Pearson Education, Inc.

Seitz, T. (2008). *Representing urban cultural geography in stabilization operations: Analysis of a social network representation in pythagoras* (Master's thesis). Monterey, California: Naval Postgraduate School.

TRADOC Analysis Center – Monterey [TRAC-MTRY]. (2009). Irregular warfare research to represent civilian populations in stability operations [PowerPoint Slides]. Retrieved from http://www.mors.org/UserFiles/file/meetings/09iw/pres/WG2_Jackson_J.pdf

U.S. Army PEO STRI. (2012). *One Semi-Automated Forces Project Office Description*. Retrieved from http://www.peostri.army.mil/PRODUCTS/ONESAF/

Watkins, C. J. C. H. & Dayan, P. (1992). Technical note: Q-Learning. *Machine Learning*, *8*, 279–292.

Watkins, C. J. C. H. (1989). *Learning from delayed rewards* (Doctoral dissertation). Retrieved from http://www.cs.rhul.ac.uk/~chrisw/

Wiedemann, M. (2010). *Robust parameter design for agent-based simulation models with application in a cultural geography model* (Master's Thesis). Monterey, California: Naval Postgraduate School.

Yamauchi, H. (2012). *Cultural geography model (version 0.9x).* Draft Technical Report. Monterey: TRAC-MTRY.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Director, TDSI
   Temasek Defence Systems Institute
   Singapore, Republic of Singapore

4. Senior Manager, TDSI
   Temasek Defence Systems Institute
   Singapore, Republic of Singapore

5. COL Lim Soon Chia
   Deputy Chief Research & Technology Officer
   Defence Research & Technology Office, Ministry of Defence
   Singapore, Republic of Singapore

APPENDIX H. SIM USER GUIDE

H.1. SIM 2.0 USER GUIDE

This Appendix contains the User Guide for SIM 2.0. The document begins by explaining how to install, compile and excute the SimKit Library-based Java code. The document continues by providing important details about building SIM 2.0 scenario files. This Appendix contains the complete User Guide in an effort to provide a complete reference for SIM Transition. This Appendix also contains changes to the scenario file necessary to run the alternate OAB method using counters. This method actually has its own User Guide; however, this appendix contains only the changes to the SIM 2.0 User Guide affected by this new methodology. The development team delivered a separate version of the User Guide for alternate OAB use to TRAC-WSMR during transition in September 2012.

Social Impact Module (SIM) Version 2.0 User Manual

TRADOC Analysis Center

29 September 2012

1 Introduction

The Social Impact Module (SIM) is a derivation of the Cultural Geography (CG) model. CG was first developed and released in 2008 as a prototype Java-based standalone tool designed to support the analysis of civilian populace behaviors during stability operations. CG was conceived to represent the population of an area of interest and their opinions on issues. CG entities perform various actions that affect and, possibly, alter the opinions of some segments of the population. These changes are recorded by the model for the analyst to examine.

CG was modified for use in the Training and Doctrine Command Analysis Center (TRAC) Irregular Warfare (IW) Tactical Wargame (TWG) held each year from 2009 to 2011. In this role, CG was used to provide feedback to the TWG players representing entities outside the population (e.g., coalition force, insurgent, host nation security force). As the TWG players initiated specific actions, CG processed the effects of these actions on the population's stances on relevant issues and their attitudes towards the TWG players.

The variant of CG developed for the TWG is now called SIM, and heretofore in this document will be referred to as SIM. This document covers SIM version 2.

2 Design and Implementation

The design of SIM follows Dr. Jacques Ferber's definition of the multi-agent system (MAS)[1]. A MAS consists of an environment, agents, objects, a set of operations that can be performed by the agents and laws that govern operations in the environment.

2.1 Environment

The SIM environment is represented by an Area of Operations (AO). Physically it is a brigade or smaller AO while logically it also contains external agents and objects influencing the agents in the AO.

2.2 Agents

The SIM agents are the actors in the simulation. The agents can represent influential individuals of society, representative family units, and groups or institutions. (Groups, institutions and other organizations are collectively called "groups" by SIM.) In the current implementation, agents are tied together based on their social network. This network is currently based on homophily.

2.3 Objects

Three types of objects are currently modeled in SIM: infrastructure, issues, and events (effects by agents in the environment that occur at a point in time and may have duration). A fourth type of object, mediums of exchange, will be implemented in the future. (Note that SIM as implemented uses the term *actions* to mean events to avoid

---

[1] J. Ferber, 1999, Multi-Agent System: An Introduction to Distributed Artificial Intelligence, Harlow: Addison Wesley Longman.

confusing a MAS event with a simulation event. This document will also use the term actions to mean MAS events.)

2.4 Operations

SIM agents perform internal and external operations. Internal operations take the form of agents processing the effects of actions and adjusting their set of beliefs, values, interests and positions on issues based on these effects. External operations take the form of agents simulating planning behaviors that allow them to initiate actions on their own. An agent affected by these actions can communicate these affects to other agents through the social network. Actions can be scripted in lieu of a behavior that has not been implemented, or be scripted to supplement an existing behavior.

2.5 Laws

Laws mediate interactions between agents and between an agent and an object. SIM implements laws through umpires. There are currently six umpires implemented called the SimpleActionUmpire, SimpleInfrastructureUmpire, SimpleDamageUmpire, SimpleLocationUmpire, SimpleSocialNetworkUmpire and PerceptUmpire.

2.5.1 SimpleActionUmpire

This umpire performs two tasks. First, the umpire determines which agent(s) receive the effects of an action when it is first executed. The agents that receive these effects will adjust their set of beliefs, values, interests and positions on issues. Second, the umpire determines whether an agent, after processing an action's effects, will be able to pass these effects on to one or more agents in its social network.

2.5.2 SimpleInfrastructureUmpire

This umpire determines which infrastructure will receive and serve agents that need to restock consumable items.

2.5.3 SimpleDamageUmpire

This umpire handles the assessment of damage and repair to infrastructure. Damage and repair are assessed in terms of changing the state of an infrastructure to, respectively, reduce or increase in the infrastructure's operational characteristics (e.g., number of servers, queue capacity, transfer rates, etc.). Repair includes renovating an undamaged infrastructure to improve its operational capacity.

2.5.4 SimpleLocationUmpire

This umpire currently assigns Agents to locations at the start of each replication. In the future, this umpire may be used to handle the movement of agents who become refugees.

2.5.5 SimpleHomophilyNetworkUmpire

This umpire currently handles the updating of link weights[2] for every pair of agents representing the civilian population. A link weight for a pair of agents measures their

[2] S. Lieberman, "Some Next Steps for Social Networks in the Cultural Geography Model", working paper dated 2009 Sep 01

similarity on a scale from 0 to 1. If the link weight is equal to 1, the agents are identical across all social dimensions; if the link weight is 0, they are the most dissimilar agents in the scenario. The link weight is used to determine whether one agent in the agent pair will communicate with the other.

2.5.6 PerceptUmpire

This umpire listens for information from the environment, forms a percept around the information, and sends the percept to the appropriate agent. In SIM, a percept contains information about the environment at a given point in time. The information is usually about an action some agent took. The information includes the time the action occurred, where it originated, and who was responsible for the action. A percept can be self-formed, i.e., the agent responsible for the action will be the agent that receives the resulting percept. An example of this occurs as an agent consumes an item and the level of that item held by the agent reaches a threshold. That agent will specify to the PerceptUmpire that is has reached the threshold with the particular item, the umpire will create a percept (in this case it will instantiate a RestockCheckPercept), and send the percept to the agent. The agent will take this percept and, based on motivational factors, it may eventually take action to address this situation, i.e., restock the item.

3 Simulation Engine

SIM is an event-stepped, stochastic model that uses Simkit 1.3.7 as its simulation engine. Simkit is an Open Source package written in Java that provides an Application Programmer Interface (API) to create discrete event simulations[3].

4 Bayesian Networks

Each agent maintains one or more Bayesian networks for updating their set of beliefs, values, interests, and positions. Each agent maintains a Bayesian network for each type of behavior that is simulated. All networks are implemented using the TRAC-MTRY developed TracBayes API (Application Programmer's Interface). This API allows SIM to operate any of the following Bayesian network packages:

- Netica-J API Library, version 4.03. This API is a commercial product developed by Norsys Software Corp. This library requires a site license and a license password to obtain the full functionality of the library.
- Weka 3. This is an open source package consisting of a collection of machine learning algorithms geared towards data mining tasks. It was developed by The University of Waikato, New Zealand and is issued under the GNU General Public License.
- LightBayes. This is an open source package developed by TRAC-MTRY.

5 Source Code

SIM may be obtained by downloading and compiling the source code.

5.1 Obtaining Source Code

[3] Simkit home page: http://diana.nps.edu/simkit/

The SIM source code is maintained in a Subversion repository. The code may be downloaded from https://soteria.nps.navy.mil/WebSVN/.

Log-on as "guest". No password is required. This provides read-only access to the repository. On the home page click on 'RUCG" and then click on "tags/". Scroll to "SIM_V_2_0_1/" that appears under the "Path" column and click on "Donwload" to the right. This will generate a compressed tar file called RUCG-SIM_V_2_0_1.rX*XXX*.tar.gz where X*XXX* is the revision number. The current size of this file should be over 17 Mb. Extract the contents of this file to a directory that can be conveniently accessed.

5.1.1 The Home Directory and Subdirectories

The SIM home directory will be the top-level directory named SIM_V_2_0_1.rX*XXX* where X*XXX* is the revision number. The home directory can be renamed for convenience; however, do not move or rename any other file or directory within or below the home directory. There are several files and subdirectories within the home directory. The most significant files and directories are described below:

a) *build.xml* – This is an Apache Ant buildfile. (See 5.2, "Compiling Source Code".)
b) *sim2.bat* – This is an example of a Microsoft Windows batch file for launching SIM. It can be modified to run any SIM scenario on a standalone workstation. (See Appendix A, "Sample Script to Launch SIM".)
c) *convert*.bat – This is an example batch file for converting a scenario spreadsheet or database to XML using SIM. It can be modified for any standalone workstation. The script is presented in Appendix B, "Sample Script to Convert Scenario Workbook to XML".
d) *logging.properties* – This is a configuration file for controlling the logging facilities.
e) *src* - This directory contains all source code of the model.
f) *tests* - This directory contains all unit test source code.
g) *lib* - This directory contains the following jar files needed to compile and run SIM:

  - coordconv.jar – coordinate conversion utility,
  - gt-geometry-2.7.1.jar – GeoTools geometry module,
  - hsqldb.jar – Open Office,
  - jdom.jar and jdom-contrib.jar – XML reader and writer
  - jtds-1.2.5.jar – jTDS JDBC driver
  - junit.jar – JUnit testing framework,
  - NpsTracCommon.jar – utilities and tools commonly used in TRAC-MTRY-developed software,
  - sim2.jar – SIM version 2
  - simkit.jar – Simkit,
  - tracBayes.jar – TracBayes API (includes LightBayes),
  - weka.jar – Weka
  - cobertura\cobertura.jar – Cobertura test coverage measurement tool

h) *data* – This directory is provided to store input files for a SIM run. There is one sample input file in XML format in the subdirectory .*mas*: example.xml. Appendix A shows how to run SIM with the sim2.bat batch file described above using this .xml input file.

Note: SIM v. 2 is not backwards compatible with input data formatted for earlier versions of SIM. See 7, "Data Inputs", for the current data requirements.

i) *output* – This directory is simply provided for convenience to store output generated from a SIM run.
j) *nbproject* - This directory contains the NetBeans IDE (integrated development environment) project files for SIM. Do not manually edit these files. NetBeans can be downloaded from http://netbeans.org/.

5.2 Compiling Source Code

The following must be installed to compile the SIM code:

- Apache Ant 1.7.1 or later (download from http://ant.apache.org/).
- Java 1.6 JDK or later (download from http://www.oracle.com/technetwork/java/javase/downloads/index.html).

After installing Ant and Java there will be two environment variables that will need to be defined, ANT_HOME and JAVA_HOME, and a third environment variable, PATH, will need to be extended. The following values assumes that SIM will be run on a 32-bit Windows XP/Vista system and that Ant 1.7.1 and Java 1.6.0_32 have already been installed in that system's C:\Program Files directory:

Variable	Value
ANT_HOME	`C:\Program Files\apache-ant-1.7.1`
JAVA_HOME	`C:\Program Files\Java\jdk1.6.0_32`
PATH	Add the following to the existing PATH: `;%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;%ANT_HOME%\bin`

Modify these values based on the system that SIM will actually be run from.

The source code download includes an Ant buildfile, build.xml, located in the SIM home directory. The code can be compiled from the command line by going to the SIM home directory and typing:

```
ant compile-test
```

The following commands are available:

Command	Description
`ant compile-test`	Creates a *build* directory (if it doesn't exist) and compiles the code in the *src* and *tests* directories, placing the generated .class

	files in the *build* directory
`ant compile`	Creates a *build* directory (if it doesn't exist) and compiles the code in the *src* directory only, placing the generated .class files in the *build* directory
`ant javadoc`	Creates a *dist* directory (if it doesn't exist) and generates Javadocs, placing them in the *dist* directory.
`ant clTest`	Runs all unit tests
`ant clean`	Removes the *build* and *dist* directories.

6 Running SIM

SIM is currently launched from the command line with the java application launcher with one, two or three arguments passed to the main method (in the class rucg.mas.main.RucgMain) as discussed below.

6.1 Running a Scenario (One Argument)

Running a scenario requires passing one argument - the name of the file containing the scenario data. The command line entry would be as follows:

```
java [options] rucg.mas.main.RucgMain inputfile
```

- *`options`* are command line options for the java application launcher. Commonly used options are illustrated in Appendix A, "Sample Script to Launch SIM".
- *`inputfile`* is the name of the input spreadsheet (.xls, .xlsb, .xlsm, .xlsx), database (.accdb, .mdb, .odb) or XML file including the path. All input case files and Bayesian network files referenced in *`inputfile`* must be in the same directory where *`inputfile`* is located. (See 7.3, "Excel/Access/Open Office Inputs", for a description of the format of *`inputfile`*.)

Note: SIM has successfully run with XML files on Linux, 32-bit Windows XP and 64-bit Windows Vista/Windows 7 systems. Assuming the appropriate ODBC drivers have been installed, SIM can read Microsoft Office 2003 Excel (.xls) and Access (.mdb) files on 32-bit Windows XP systems; and Microsoft Office3 2007/2010 Excel (.xlsb, .xlsm, .xlsx) and Access (.accdb) on 32-bit Windows XP and 64-bit Windows Vista/Windows 7 systems. The Open Office database has only been unit tested on 32-bit Windows XP and 64-bit Windows Vista/Windows 7 systems.

6.2 Converting to XML (Two or Three Arguments)

When a scenario is run with an input spreadsheet or database as specified in the previous section, SIM first converts the input spreadsheet/database to XML and then reads the resulting XML file before proceeding with the run. There are times when the user may only want to convert to XML and avoid running the scenario. This can be done by passing one or two additional arguments as explained below.

6.2.1 Conversion Using Two Arguments

The most common method that will allow SIM to convert to XML without running the scenario involves supplying two arguments as follows:

```
java [options] rucg.mas.main.RucgMain inputfile convert
```

- *options* are command line options for the java application launcher.
- *inputfile* is the name of the input spreadsheet (.xls, .xlsb, .xlsm, .xlsx) or database (.accdb, .mdb, .odb) to be converted. The name includes the path. All input case files and Bayesian network files referenced in *inputfile* must be in the same directory where *inputfile* is located.
- `convert` is entered as is and instructs SIM to convert *inputfile* to XML without running the scenario.

The resulting XML file will begin with the same name as *inputfile* and will be written to the same directory.

6.2.2 Conversion Using Three Arguments

By default, the two-argument conversion will prefix any case file or Bayesian network file listed in the input spreadsheet/database with its path when it is written out to XML. This is fine when SIM is run on a standalone workstation, however, when SIM is run on a high performance computing cluster (see 6.3), the path needs to be left out. To leave out the path, provide an additional argument as follows:

```
java [options] rucg.mas.main.RucgMain inputfile convert noInputPath
```

- *options* are command line options for the java application launcher.
- *inputfile* is the name of the input spreadsheet (.xls, .xlsb, .xlsm, .xlsx) or database (.accdb, .mdb, .odb) to be converted. The name includes the path. All input case files and Bayesian network files referenced in *inputfile* must be in the same directory where *inputfile* is located.
- `convert` is entered as is and instructs SIM to convert *inputfile* to XML without running the scenario.
- `noInputPath` is entered as is and instructs SIM to not include the path when it writes the name of a case or Bayesian network file to XML.

The resulting XML file will begin with the same name as *inputfile* and will be written to the same directory.

6.3 Running on a Cluster (Three Arguments)

To enable SIM to run on a high performance computing cluster, three arguments are supplied as follows:

```
java [options] rucg.mas.main.RucgMain inputxml inputdirectory
outputdirectory
```

- *options* are command line options for the java application launcher.

- *inputxml* is the name of the input XML file including the path.
- *inputdirectory* is the path to the directory containing the input case files and Bayesian network files referenced in *inputxml*.
- *outputdirectory* is the path to the directory that the data loggers will write their output to.

Note: this command line entry can be used to run SIM on a standalone workstation, but the user must insure that the names of the case files and Bayesian network files listed in *inputxml* DO NOT include the path to these files.

7 Data Inputs

Data can be read from an XML file, a Microsoft Excel spreadsheet, a Microsoft Access database, or an Open Office database. Excel 2003 (*.xls) and 2007/2010 (*.xlsx, *.xlsm, *.xlsb) spreadsheets, and Access 2003 (*.mdb) and 2007/2010 (*.accdb) databases are supported.

7.1 Time Units

The unit of time in a SIM scenario is arbitrary and is left for the analyst to choose. Since there are many data inputs that require a rate, a starting time, or time interval, it is important that these inputs are consistent with the chosen time unit.

7.2 Data Types

In each of the tables that follow in 7.3, “Excel/Access/Open Office Inputs”, the type of data required for the input is described in a column called “Type”. The data type is expressed either as a Java primitive type or a class and the possible types are defined as follows:

Type	Description	Note
int	32-bit signed integer	
long	64 bit signed integer	
double	64-bit floating point	
String	Text string	All String inputs are case sensitive.

7.2.1 Specifying Distributions

There are inputs that require the name of a distribution to be entered. These inputs are generally associated with specifying an execution time, or a time interval, and are entered as Strings. The format requires the name of the distribution to be provided followed by the parameter(s) of the distribution within parentheses. The name of the distribution is the class name of a distribution defined in Simkit’s simkit.random package or a java class implementing the simkit.random.RandomVariate interface. A sampling of continuous distributions available from the simkit.random package is presented below. See the Simkit javadocs for details of these classes.

Class Name	Short Name	# of Parameters

BetaVariate	Beta	2
ConstantVariate	Constant	1
ExponentialVariate	Exponential	1
GammaVariate	Gamma	2
InverseGaussianVariate	InverseGaussian	2
LogNormalVariate	LogNormal	2
NormalVariate	Normal	2
Normal02Variate	Normal02	2
Normal03Variate	Normal03	2
TriangleVariate	Triangle	3
UniformVariate	Uniform	2
WeibullVariate	Weibull	2

To specify a distribution, enter the short name of the distribution followed by a parameter list within parentheses. For two- and three-parameter distributions, each parameter must be separated by a comma. Examples of one, two, and three parameter distribution inputs are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

The distributions listed above can be used when SIM is run on 32-bit systems. Care must be taken, however, when SIM is run on 64-bit systems. Distributions that call Java's Math.log() method may not return consistent values on 64-bit systems. Simkit provides an alternative natural log algorithm that provides consistent results on 64-bit systems and, for those distributions that require natural logarithms to be calculated, this alternative algorithm is called in 64-bit versions of the distributions listed above. These 64-bit distributions should be used in order to avoid replication problems on 64-bit systems. The previous 32-bit distributions and their 64-bit counterparts are listed below.

Distribution for 32-Bit Systems	Equivalent Distribution for 64-Bit Systems
BetaVariate	None
ConstantVariate	ConstantVariate
ExponentialVariate	Exponential_64Variate
GammaVariate	Gamma_64Variate
InverseGaussianVariate	None
LogNormalVariate	None
NormalVariate	NormalVariate_64
Normal02Variate	Normal02_64Variate
Normal03Variate	Normal03_64Variate
TriangleVariate	TriangleVariate
UniformVariate	UniformVariate
WeibullVariate	Weibull_64Variate

7.3 Excel/Access/Open Office Inputs

The format of the input is the same whether the data resides in Excel, Access, or Open Office. The spellings of the worksheet/table names and column/field names are significant and must match the names that appear in Tables 1 through 48. The data are entered in 48 worksheets/tables and are described as follows:

7.3.1 ScenarioData Worksheet/Table

Provides information for controlling the run. Only one row of data is required. The data is described in Table 7.3.1.

Column/Field Name	Type	Description
ScenarioLength	double	The length of the scenario in arbitrary time units.
Replications	int	The number of replications to run.
verbose	String	Indicates whether the event list should be shown after each event is executed. Used only for debugging, therefore “FALSE” should normally be entered.
reallyVerbose	String	Indicates whether additional debug/trace information should be shown. Used only for debugging, therefore “FALSE” should normally be entered.

Table 7.3.1 ScenarioData Worksheet/Table.

7.3.2 Seeds Worksheet/Table

Specifies the seeds for the random number generator for each replication. One row must be filled out for each replication as shown in Table 7.3.2.

Column/Field Name	Type	Description
replication	int	The replication number.
seed	long	The seed value for the replication.

Table 7.3.2 Seeds Worksheet/Table.

7.3.3 BeliefPrototype Worksheet/Table

Defines the beliefs/interests/values held by agents based on the scenario. These are called BeliefPrototypes and they affect an agent’s position on issues. Each row of data defines a BeliefPrototype as shown in Table 7.3.3.

Column/Field Name	Type	Description
name	String	The name of the belief. The belief name must match the name of a node representing a belief in a Bayesian network file declared in Table 7.3.23.
shortDescripiton	String	A brief description of this belief.
longDescription	String	A more detailed description of this belief.

Table 7.3.3 BeliefPrototype Worksheet/Table.

7.3.4 BeliefPositionPrototype Worksheet/Table

Each belief in Table 7.3.3 has a set of positions. One row is filled out for each position as shown in Table 7.3.4.

Column/Field Name	Type	Description
name	String	The name of the belief position. The position name must match a state name of a node representing a belief in a Bayesian network file declared in Table 7.3.23.
beliefPrototype	String	The name of a BeliefPrototype defined in Table 7.3.3.
description	String	A description of this position.

Table 7.3.4 BeliefPositionPrototype Worksheet/Table.

7.3.5 IssuePrototype Worksheet/Table

Defines the issues important to the agents based on the scenario. These are called IssuePrototypes and each row of data defines an IssuePrototype as shown in Table 7.3.5.

Column/Field Name	Type	Description
name	String	The name of the issue. The issue name must match the name of the node representing an issue in a Bayesian network file declared in Table 7.3.23.
shortDescripiton	String	A brief description of this issue.
longDescription	String	A more detailed description of this issue.

Table 7.3.5 IssuePrototype Worksheet/Table.

7.3.6 IssuePositionPrototype Worksheet/Table

Each issue in Table 7.3.5 has a set of positions. One row is filled out for each position as shown in Table 7.3.6.

Column/Field Name	Type	Description
name	String	The name of the issue position. The position name must match a state name of a node representing an issue in a Bayesian network file declared in Table 7.3.23.
issuePrototype	String	The name of an IssuePrototype defined in Table 7.3.5.
description	String	A description of this position.

Table 7.3.6 IssuePositionPrototype Worksheet/Table.

7.3.7 AttitudePrototype Worksheet/Table

An AttitudePrototype defines an attitude that population agents display towards an AgentPrototype representing an external player (see 7.3.15 “AgentPrototype Worksheet/Table”). Examples of external players are coalition forces, the host nation government, NGOs, mass media, and insurgents. Each row of data defines an AttitudePrototype as shown in Table 7.3.7.

Column/Field Name	Type	Description

name	String	The name of the attitude. The attitude name must match the name of the node representing an attitude in a Bayesian network file declared in Table 7.3.23.
attitudeTowards	String	The name of the AgentPrototype defined in Table 7.3.15 that this is attitude is directed towards. (See Note.)

Table 7.3.7 AttitudePrototype Worksheet/Table.

Note: The “isExternal” field of the AgentPrototype (see Table 7.3.15) should be “TRUE”.

7.3.8 AttitudePositionPrototype Worksheet/Table

Each attitude in Table 7.3.7 has a set of positions. One row is filled out for each position as shown in Table 7.3.8

Column/Field Name	Type	Description
name	String	The name of the attitude position. The position name must match a state name of a node representing an attitude in a Bayesian network file declared in Table 7.3.23.
attitudePrototype	String	The name of an AttitudePrototype defined in Table 7.3.7.
category	String	Enter “POSITIVE”, “NEUTRAL” or “NEGATIVE” if this position can be classified as a positive, neutral or negative attitude, respectively.
paveField	String	The name of the column in PAVE’s CG_ObservedAttitude table that corresponds to this position. (See Notes 1 and 2.)

Table 7.3.8 AttitudePositionPrototype Worksheet/Table.

Note 1: The column names currently are "CGposActiveResponse", "CGposPassiveResponse", "CGdoNothing", "CGnegPassiveResponse" and "CGnegActiveResponse".

Note 2: The “paveField” field is only needed when SIM must interface with PAVE (Planning, Adjudication, and Visualization Environment). Enter “NA” if SIM does not have to interface with PAVE.

7.3.9 SocialDimension Worksheet/Table

Defines the social dimensions over which link weights for each agent pair will be calculated. Each row of data defines a SocialDimension as shown in Table 7.3.9.

The social dimension may be either a static ascribed characteristic such as tribe, education, political affiliation or age; or it may be a belief, value, interest or issue in which the stances/positions may change during a simulation run. Although in reality the static dimensions may move over time, SIM considers them fixed throughout a run.

Each dimension must be assigned a weight that indicates the relative importance of that dimension. If d dimensions are defined in this table and c_i is the weight assigned to dimension i, each c_i must obey the following constraints: $0 \le c_i \le 1 \forall i \in (1,...,d)$ and $\sum_{i=1}^{d} c_i = 1$.

Column/Field Name	Type	Description
name	String	The name of the social dimension. If this dimension is dynamic, the name must be that of a BeliefPrototype declared in Table 7.3.3 or an IssuePrototype declared in Table 7.3.5.
homophilyWeight	double	The relative importance of this dimension in the range [0.0, 1.0]. The sum of these weights over all of the dimensions must add to 1.0.

Table 7.3.9 SocialDimension Worksheet/Table

7.3.10 SocialDimensionValueType Worksheet/Table

Defines the classifications/categories within each social dimension and assigns a numerical value to each one. The numerical value defines the position that that category occupies along the dimension. Each row of data defines a category within a SocialDimension as shown in Table 7.3.10. Note that category names need to be unique within a social dimension but may be used repeatedly between different social dimensions.

Column/Field Name	Type	Description
category	String	The name of the classification/category of a static SocialDimension or the position/stance of a BeliefPrototype or IssuePrototype.
socialDimension	String	The name of a SocialDimension defined in Table 7.3.9 that category is assigned to.
value	double	The value must be greater than or equal to 0.

Table 7.3.10 SocialDimensionValueType Worksheet/Table.

The values assigned to each category in a given dimension are used to determine the distance between agents in that dimension. The values must be on the same scale within a dimension, but different dimensions may use different scales. For example, a five-point Likert scale may be used on one dimension while a 0-100 socioeconomic index may be used on another dimension. The distances between agents in a given dimension will be normalized by the maximum distance between any two agents in that dimension, resulting in a scalar between 0 and 1. Once the distances have been normalized for each dimension, they can be combined to calculate the social distance between two agents which, in turn, can be used to calculate the link weight of the agent pair[4].

Example: Suppose there is a dimension called "Disposition" that classifies agents in the civilian population as either "Rural" or "Urban". Suppose Rural is assigned a value of 1 and Urban is assigned a value of 2. The distance between a Rural agent and an Urban agent will be 1 within this dimension while the distance between two Rural agents or two Urban agents will be 0.

7.3.11 CognitiveArchitecture Worksheet/Table

A single row of data must be entered that covers an assortment of parameters required by the Cognitive Architecture. The entries are described in Table 7.3.11.

Column/Field Name	Type	Description
selectiveAttentionThreshold	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the attention threshold for each agent. (See Note 1.)
workingMemoryCapacity	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the working memory capacity for each agent. (See Note 2.)
expectedCommunication	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the expected number of times an agent (from the civilian population) will communicate with another civilian agent over a set time period, specified by *expectedCommunicationTimeUnits*. (See Notes 2 and 3.)
expectedCommunicationTimeUnits	double	The time period over which the number of times each civilian agent communicates is tracked.
experienceThreshold	int	Determines whether an agent has enough experience. (See Note 4.)

[4] S. Lieberman, "Some Next Steps for Social Networks in the Cultural Geography Model", working paper dated 2009 Sep 01

volatilityThreshold	double	The threshold that indicates, based on recent actions, whether an agent has tended to select actions that result in consistent rewards over actions that result in uneven (volatile) rewards. (See Note 5.)
volatilityPeriods	int	The number of time periods over which volatility is measured. (See Note 5.)
volatilityPeriodLength	double	The length of each time period over which volatility is measured (See Note 5.)
physiologicalWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0, 1.0]. This weight is applied to the motivation score for immediate physiological needs. (See Note 6.)
selfProtectionWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0, 1.0]. This weigh is applied to the motivation score for self-protection. (See Note 6.)
affiliationWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0, 1.0]. This weight is applied to the motivation score for affiliation. (See Note 6.)
statusEsteemWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0, 1.0]. This weight is applied to the motivation score for status/esteem. (See Note 6.)
temperature	double	The temperature for generating the Boltzmann distribution in the range [0.0, ∞). Used during the MetaCognition process to determine goals.
filterTrust	String	Enter “TRUE” or “FALSE” if trust filtering will be on or off, respectively. (See Note 7.)
effectsLambda	double	The discount rate for calculating the effects of actions (scripted and behavioral) in the range [0.0, 1.0).

Table 7.3.11 CognitiveArchitecture Worksheet/Table.

Note 1: The attention threshold is used to determine whether a percept should be added to working memory. If the age of the percept (time percept received minus time percept was formed) is less than the attention threshold, the percept is added to working memory; otherwise, it is discarded. SIM will throw a RuntimeException if the distribution generates a value that is less than or equal to zero.

Note 2: Since the simkit.random.RandomVariate.generate() method returns a double, the result will be rounded to the nearest integer. If the value after rounding is less than zero, SIM will throw a RuntimeException. If the value is zero, the meta-cognition process will handle this by setting the agent's motivation score for sending communication to zero. The agent will still be allowed to receive communication sent by another agent.

Note 3: Currently communication is only considered between agents in the civilian population. Informant communication (where a civilian agent provides information to an agent representing an external player, such as a coalition force agent) is being considered for future implementation.

Note 4: Experience is defined by the number of trials of each action taken. The agent tracks the number of times each action was taken. If the number of times action X was taken is less than or equal to the value held in the "experienceThreshold" field, the agent is considered to have insufficient experience with regard to Action X; otherwise, the agent is considered to have sufficient experience. Experience is one of two factors used during the ActionSelection process to choose a decision method: exploration learning, recognition prime decision making (RPD), or mental stimulation.

Note 5: Volatility is a measure of risk. It is the second of two factors used during the ActionSelection process to choose a decision method. Volatility is measured over a set number of time periods ("volatilityPeriods") of specified length ("volatilityPeriodLength"). For each action taken over these time periods, the maximum and minimum expected utilities resulting from these actions are tracked. For a given action in a given time period, the ratio of the maximum expected utility over the minimum expected utility yields the volatility of that action in that time period. The maximum volatility is the maximum volatility over all actions and all time periods. If the maximum volatility exceeds a specified threshold ("volatilityThreshold"), the volatility is considered high (more risk); otherwise the volatility is considered low (less risk).

Note 6: The sum of "physiologicalWeight", "selfProtectionWeight", "affiliationWeight" and "statusEsteemWeight" must add to 1.0.

Note 7: If trust filtering is on, an agent will only communicate with the agents in its social network that it trusts, and an agent that receives information from another agent will accept that information only if it trusts the sender; otherwise, the information is ignored. If trust filtering is off, an agent will always communicate with the agents in its social network, and it will always accept information that it receives from the other agents in its social network.

7.3.12 IssueSatisfactionType Worksheet/Table

During the MetaCognition process an agent performs a cognitive appraisal where a satisfaction value in the range [0.0, 1.0] is calculated. The larger this value, the more "satisfied" the agent is with the current state of affairs. The calculation of satisfaction consists of two components: motivation scores and issue stances. This table identifies the issue(s) that will contribute to the calculation.

Column/Field Name	Type	Description
name	String	The name for grouping the issues. This name will be referenced by an AgentPrototype listed in Table 7.3.15.
issue	String	The name of the IssuePrototype defined in Table 7.3.5. This is the issue that is considered relevant for calculating satisfaction.
position	String	The name of the IssuePositionPrototype defined in Table 7.3.6. This is *issue's* position whose probability or likelihood will be incorporated into the calculation of satisfaction.
weight	double	The weight that *issue* contributes in the range [0.0, 1.0]. The weights of all issues assigned to *name* must sum to 1.0.

Table 7.3.12 IssueSatisfactionType Worksheet/Table

Each group of issues identified by the "name" field can be tailored to one or more AgentPrototypes (see 7.3.15 "AgentPrototype Worksheet/Table"). In this way, one set of issues and positions can be created that are appropriate for, say, "passive" agents while another set of issues and positions can be created for agents that are "radical".

7.3.13 PerceptUmpire Worksheet/Table

Defines the PerceptUmpire. Only one row of data is required. The data is described in Table 7.3.13.

Column/Field Name	Type	Description
name	String	The name of the umpire.
class	String	The class name of a PerceptUmpire defined in the rucg.mas.behavior.cognitive package. (See Note.)

Table 7.3.13 PerceptUmpire Worksheet/Table

Note: Enter either "CgPerceptUmpire" or "rucg.mas.behavior.cognitive.CgPerceptUmpire" (without the quotes).

7.3.14 ConsumableType Worksheet/Table

Defines the types of goods and services consumed by agents and stored at infrastructures. Each row of data defines a ConsumableType as shown in Table 7.3.14.

Column/Field Name	Type	Description
name	String	The name of the type of consumable.

Table 7.3.14 ConsumableType Worksheet/Table.

7.3.15 AgentPrototype Worksheet/Table

Agents are classified by AgentPrototype. Each row of data defines an AgentPrototype as shown in Table 7.3.15.

Column/Field Name	Type	Description
name	String	The name of the prototype.
isGroup	String	If this prototype represents a group, organization or institution, enter “TRUE”; otherwise, enter “FALSE”.
isExternal	String	If this prototype represents an external player, enter “TRUE”. Enter “FALSE” if this prototype represents a stereotype of the civilian population. (See Note 2.)
isMedia	String	If *isGroup* is “TRUE” and this prototype represents the mass media, enter “TRUE”; otherwise, enter “FALSE”.
moveRate	String	If *isExternal* is “FALSE”, enter the distribution for generating a movement rate when an agent of this type needs to travel to and from an infrastructure to obtain goods or services. (See Notes 1 and 3) Ignored if *isExternal* is “TRUE”.
issueSatisfactionType	String	If *isExternal* is “TRUE”, enter the name of the IssueSatisfactionType defined in Table 7.3.12. If not applicable, enter “NA”. Ignored if *isExternal* is “FALSE”. (See Note 4.)

Table 7.3.15 AgentPrototype Worksheet/Table.

Note 1: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 2: Examples of external players are coalition forces, the host nation government, NGOs, mass media, and insurgents.

Note 3: Movement rate distributions should be entered if the “spatialMethod” field of the SimpleInfrastructureUmpire defined in Table 7.3.54 is “PROXIMITY”. Each distribution should generate a rate measured in kilometers per unit time if coordinates of Locations (see 7.3.32 “Location Worksheet/Table”) are GEODETIC, MILGRID or UTM. If Locations use ARBITRARY_X_Y coordinates, however, the distance is measured in an arbitrary unit consistent with that coordinate system. If the “spatialMethod” field of the SimpleInfrastructureUmpire is “COLLOCATION”, the “delayClass” field in table 7.3.53 should be used instead and “NA” should be entered in the “moveRate” field.

Note 4: When an agent performs a cognitive appraisal it calculates a satisfaction value. This value is in the range [0.0, 1.0] and the larger the value, the more “satisfied” the agent is with the current state of affairs. The calculation of satisfaction consists of two components: motivation scores and issue stances. The “issueSatisfactionType” field addresses the issue stance component and identifies the group of issues that are relevant to calculating the satisfaction. These groups were specified in Table 7.3.12 “IssueSatisfactionType Worksheet/Table”. If “NA” is entered in the field, the satisfaction calculation will only consider motivation scores. Note that external players do not evaluate issues; therefore, they will base their satisfaction on motivation scores only.

7.3.16 AgentSocialDimensions Worksheet/Table

Assigns to each AgentPrototype a category from each static SocialDimension that best characterizes the prototype in that dimension. One row is filled out for each static dimension for each prototype as shown in Table 7.3.16. External agent prototypes only have SocialDimensions if they are participating in the dynamic social network.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.15.
socialDimension	String	The name of a **static** SocialDimension defined in Table 7.3.9.
Category	String	The name of a category that is assigned to *socialDimension* in Table 7.3.10.

Table 7.3.16 AgentSocialDimensions Worksheet/Table.

7.3.17 AgentBeliefs Worksheet/Table

Defines beliefs/interests/values held by AgentPrototype. One row is filled out for each belief held by an AgentPrototype as shown in Table 7.3.17.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.15 whose “isExternal” field is “FALSE”.
beliefPrototype	String	The name of the BeliefPrototype defined in Table 7.3.3.

Table 7.3.17 AgentBeliefs Worksheet/Table.

7.3.18 AgentIssues Worksheet/Table

Defines issues important to AgentPrototype. One row is filled out for each issue important to an AgentPrototype as shown in Table 7.3.18.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.15 whose “isExternal” field is “FALSE”.
issuePrototype	String	The name of the IssuePrototype defined in Table 7.3.5.

Table 7.3.18 AgentIssues Worksheet/Table.

7.3.19 AgentAttitudes Worksheet/Table

Defines attitudes held by AgentPrototype. One row is filled out for each attitude held by an AgentPrototype as shown in Table 7.3.19.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.15 whose “isExternal” field is “FALSE”.
attitudePrototype	String	The name of the AttitudePrototype defined in Table 7.3.7.

Table 7.3.19 AgentAttitudes Worksheet/Table

7.3.20 Agent Worksheet/Table

Defines each agent to be instantiated in the scenario. One row of data is entered for each agent as shown in Table 7.3.20.

Column/Field Name	Type	Description
name	String	The name of the agent.
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.15.
initialLocation	String	Either the name of the Location defined in Table 7.3.32 where this agent will be initially located (i.e., at time 0), or “ANY”. If the latter, the SimpleLocationUmpire will assign an initial Location to this agent.
keyLeader	Boolean	True if this agent is a key leader in the scenario.
trustFraction	double	The fraction of nearest K neighbors this agent will choose to communicate with, in the range [0.0, 1.0]. (See Note 1.)
trustTemperature	double	The temperature at which this agent chooses its trustworthy agents, in the range (0.0, ∞). (See Notes 1 and 2.)
defaultTrust	double	The default (initial) trust value this agent uses when it has no trust value about another agent. (See Note 1.)
lambdaSend	double	(See Note 1.)
gammaOrOneSend	double	(See Note 1.)
exploreModeSend	String	Enter “BOLTZMANN” or “EPSILON_GREEDY” (See Note 1.)
epsilonOrTemperatureSend	double	(See Note 1.)
defaultUtilitySend	double	(See Note 1.)
modeSend	String	Enter “Q_LEARNING”, “SARSA” or “DIRECT_Q_COMPUTATION” (See Note 1.)
lambdaReceive	double	(See Note 1.)

gammaOrOneReceive	double	(See Note 1.)
exploreModeReceive	String	Enter “BOLTZMANN” or “EPSILON_GREEDY” (See Note 1.)
epsilonOrTemperatureReceive	double	(See Note 1.)
defaultUtilityReceive	double	(See Note 1.)
modeReceive	String	Enter “Q_LEARNING”, “SARSA” or “DIRECT_Q_COMPUTATION” (See Note 1.)

Table 7.3.20 Agent Worksheet/Table.

Note 1: Ignored if the “filterTrust” field in Table 7.3.11 is “FALSE”, i.e., trust filtering is turned off.

Note 2: Higher temperature indicates more random behavior, whereas lower temperature indicates stricter adherence to trust values.

7.3.21 GroupMembers Worksheet/Table

Lists members of groups, organizations and institutions. One row of data is entered for each member as shown in Table 7.3.21.

Column/Field Name	Type	Description
name	String	The name of the agent defined in Table 7.3.20 that represents a group, organization or institution, i.e., the “isGroup” field of its AgentPrototype (see Table 7.3.15) is “TRUE”.
member	String	The name of an agent defined in Table 7.3.20 that is a member of *name*.

Table 7.3.21 GroupMembers Worksheet/Table.

7.3.22 CaseFiles Worksheet/Table

Lists all of the case files that will be used to initialize and update the Bayesian networks described in 7.3.23, “BayesNetFiles Worksheet/Table” and 7.3.25, “Behavior Worksheet/Table”. One row is entered for each case file as described in Table 7.3.22.

All of the case files must reside in the same directory as the spreadsheet or database file. They are assumed to be in comma-delimited format (*.csv).

Column/Field Name	Type	Description
caseFile	String	The name of the case file. Do not enter the full path name of the case file.

Table 7.3.22 CaseFiles Worksheet/Table.

7.3.23 BayesNetFiles Worksheet/Table

Maps IssuePrototypes defined in Table 7.3.5 to Bayesian networks used to evaluate positions on issues. Each row defines a map between a network file and either an IssuePrototype or AttitudePrototype. The entries are described in Table 7.3.23.

The network file only contains the structure of the Bayesian network; the initial state of the network is set by the AgentNets worksheet described in 7.3.24 “AgentNets Worksheet/Table”.

Column/Field Name	Type	Description
class	String	Enter “IssueNet” or “AttitudeNet” depending upon whether *bayesNetFile* is built around issues or atttiudes, respectively. (See Note 1.)
bayesNetFile	String	The name of the Bayesian network file holding a network structure built around a one or more issues and a set of supporting beliefs, values and interests, or the name of the Bayesian network file holding a network structure built around a one or more attitudes and a set of supporting beliefs, values and interests. (See Notes 2 and 3.)
prototype	String	The name of the IssuePrototype defined in Table 7.3.5, or the name of the AttitudePrototype defined in Table 7.3.7, or “MULTIPLE”. (See Note 4.)

Table 7.3.23 BayesNetFiles Worksheet/Table.

Note 1: The package name “rucg.mas.bayesnet” may be optionally prefixed to the class entry. Therefore, “IssueNet”, “rucg.mas.bayesnetIssueNet”, AttitudeNet and “rucg.mas.bayesnet.AttitudeNet” are legitimate entries (without the quotes).

Note 2: If SIM is running Weka or LightBayes, the Bayesian network file must be in text format following the DNET file specification (*.dne).

Note 3: All Bayesian network files must be in the same directory as the Excel, Access, or Open Office file containing the scenario data. Do not enter the full path name of the network file.

Note 4: If the file specified by *bayesNetFile* addresses only a single issue, enter the IssuePrototype name. The name must match the name of the node representing the issue. If the file addresses more than one issue, enter “MULTIPLE”. Likewise, if the file specified by *bayesNetFile* addresses only a single attitude, enter the AttitudePrototype name. The name must match the name of the node representing the attitude. If the file addresses more than one attitude, enter “MULTIPLE”.

7.3.24 AgentNets Worksheet/Table

This is used to declare the issues and attitudes that are relevant to each agent. It is also used to set the agent’s initial positions on the issues and attitudes. Each row is filled out according to Table 7.3.24.

Column/Field Name	Type	Description
agent	String	The name of the agent defined in Table 7.3.20.
issueNetFile	String	The name of the Bayesian network file declared in Table 7.3.23 for assessing positions on issues.
initialCaseFile1	String	The name of a case file defined in Table 7.3.22 or "NONE". (See Note 1). The file holds this agent's initial findings for the issue(s) represented by *issueNetFile*. This file is assumed to be in comma-delimited format (*.csv). (See Note 2.)
weight1	double	The weight applied to *initialCaseFile1*. It should have a positive value. (See Note 3.)
attitudeNetFile	String	The name of the Bayesian network file declared in Table 7.3.23 for assessing positions on attitudes.
initialCaseFile2	String	The name of a case file defined in Table 7.3.22 or "NONE". (See Note 1). The file holds this agent's initial findings for the attitude(s) represented by *attitudeNetFile*. This file is assumed to be in comma-delimited format (*.csv). (See Note 2.)
weight2	double	The weight applied to *initialCaseFile2*. It should have a positive value. (See Note 3.)
attitudeSelectCycleClass	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the time this agent waits before selecting a position on each attitude in *attitudeNetFile*. (See Note 4.)

Table 7.3.24 AgentNets Worksheet/Table.

Note 1: Enter the name of a case file if learning is required to determine the initial probabilities of the conditional probability tables in the *NetFile*. Enter "NONE" if the conditional probability tables in the *NetFile* were created manually by a subject matter expert, or have already been learned well.

Note 2: If a case file name is entered, the case file must be in the same directory as the Excel, Access, or Open Office file containing the scenario data. Do not enter the full path name of the case file.

Note 3: The weight, also called degree, represents the multiplicity of the case(s) in *initialCaseFile*. A positive value of *w* is used to tell the Bayesian network to learn *w* cases at once. A negative value of –*w* is used to tell the network to "unlearn" *w* cases at once. It is assumed, however, that if an initial case file is specified, it will be used to train the network to obtain the initial beliefs, values, interests, and positions on an issue.

Therefore, a negative weight should never be entered in this worksheet. There is no effect on the network if $w = 0$. The weight normally is 1.

Note 4: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.25 Behavior Worksheet/Table

Declares the planned behaviors. Each row defines a behavior. The data is described in Table 7.3.25.

The behaviors for each agent are stored in a Map structure keyed to the name of the behavior. Therefore, each behavior must be identified by a unique name, even if the behaviors are the same "behaviorType".

The network file only contains the structure of the Bayesian network. The structure, itself, is based on Icek Aizen's Theory of Planned Behavior[5] (See 8, "Bayesian Networks for Simulating Planned Behaviors"). The initial state of the network is set by the AgentBehaviors worksheet described in 7.3.31, "AgentBehaviors Worksheet/Table".

Column/Field Name	Type	Description
name	String	The name of the behavior.
behaviorType	String	Enter "COALITION_ACTION", "GOVERNMENT_ACTION", "INFORMANT_ACTION", "INFRASTRUCTURE", "INSURGANT_ACTION", "MASS_MEDIA", "KLE_ACTION" or "COMMUNICATE". (See Note 1.)
consumableType	String	If *behaviorType* is "INFRASTRUCTURE", the name of the ConsumableType from Table 7.3.14 that will be restocked. Ignored if *behaviorType* is not "INFRASTRUCTURE".

Table 7.3.25 Behavior Worksheet/Table.

Note 1: Behavior types provide a means to categorize the behaviors. Each type is described as follows:

a. COALITION_ACTION is a behavior whose possible actions are associated with coalition forces.
b. GOVERNMENT_ACTION is a behavior whose possible actions are associated with the host nation government.
c. INFORMANT_ACTION is a behavior whose possible actions are associated with an agent that informs another agent representing the media.

[5] Theory of Planned Behavior home page: http://people.umass.edu/aizen/tpb.html

d. INFRASTRUCTURE is a behavior whose possible actions are associated with an agent that seeks to replenish consumables from an infrastructure.
e. INSURGANT_ACTION is a behavior whose possible actions are associated with insurgents.
f. MASS_MEDIA is a behavior whose possible actions are associated with the mass media.
g. KLE_ACTION is a behavior associated with the key leader(s) of a group whose possible actions involve meeting with representatives of another group (e.g., tribal leaders meeting with coalition forces).
h. COMMUNICATE is a behavior whose possible actions are associated with a population agent communicating with another population agent in its social network.

Note 2: If SIM is running Weka or LightBayes, the Bayesian network file must be in text format following the DNET file specification (*.dne). If SIM is running NeticaJ, the file may be either in Netica's own file specification (*.neta) or the DNET format.

Note 3: All Bayesian network files must be in the same directory as the Excel, Access, or Open Office file containing the scenario data. Do not enter the full path name of the network file.

7.3.26 IntentNodeStates Worksheet/Table

For each behavior declared in Table 7.3.25, each state from the node representing the Intention to perform the behavior must be mapped to a method to be invoked by an agent. One row is filled out for each state as shown in Table 7.3.26.

Column/Field Name	Type	Description
behaviorName	String	The name of the behavior declared in Table 7.3.25.
intentNodeState	String	The name of the state from the node in *behaviorName* that represents the Intention to perform the behavior. This describes an action that the agent may perform.
execute	String	The name of the method to be executed when the agent chooses to act on *intentNodeState*. This must be "pbDoNothing", "pbCommunicate," "pbInformantCommunicate", "pbMassCommunicate", "pbDamge", "pbRepair", "pbSeek", "pbUseCurrent", or "pbKeyLeaderEngagement". (See Note 1.)
delayClass	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the delay time between the time the agent chooses this state as its course of action and the time the

		agent starts executing that action. If there is no delay, enter "Constant(0.0)". (See Note 2.)

Table 7.3.26 IntentNodeStates Worksheet/Table.

Note: The execute field must contain one of the nine entry choices listed below. Each entry is the name of a method (starting with the letters "pb") in the Java class rucg.mas.agent.Agent and its subclass rucg.mas.agent.group.Group. Because these are method names, the entries are case sensitive. SIM will throw a "NoSuchMethodException" if the method name is misspelled, or is not in the proper case (e.g., "pbdonothing" instead of "pbDoNothing").

a. pbDoNothing – This method is associated with all behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that allows the entity not to take any action at all, i.e., a "do nothing" state. This method should be entered for this state.
b. pbCommunicate - This method is associated with all behavior types. With some intention node states, the interest is not with the physical execution of the behavior, but with the effects on agents that result from that execution. This method should be entered for these states. When invoked, this method starts the communication of the effects to other agents in the social network.
c. pbInformantCommunicate – This method should only be associated with INFORMANT_ACTION behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that gives the entity the option to communicate with the media. This method should be entered for this state.
d. pbMassCommunicate – This method should only be associated with MASS_MEDIA behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that gives the entity the option to broadcast to its audience. This method should be entered for this state.
e. pbDamage – This method is generally associated with INSURGANT_ACTION behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that gives the entity the option to attack an infrastructure. This method should be entered for this state.
f. pbRepair – This method is generally associated with GOVERNMENT_ACTION behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that gives the entity the option to repair an infrastructure. This method should be entered for this state.
g. pbSeek – This method should only be associated with INFRASTRUCTURE behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that gives the agent the option to find an alternate infrastructure that can provide a needed ConsumableType (as opposed to returning to the last infrastructure that provided that ConsumableType). This method should be entered for this state.
h. pbUseCurrent – This method should only be associated with INFRASTRUCTURE behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that gives the agent the option to return to the last infrastructure that provided a needed ConsumableType (as

opposed to looking for an alternate infrastructure that can provide the same ConsumableType). This method should be entered for this state.

i. pbKeyLeaderEngagement – This method should only be associated with KLE_ACTION behavior types. The intention node of the behavior declared in the "behaviorName" field should have a state that gives the agent the option to initiate a Key Leader Engagement. This method should be entered for this state.

Due to the ongoing development of SIM, these methods may be removed, replaced, or supplemented with additional methods in future releases of SIM.

Note 2: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.27 UtilityBehavior Worksheet/Table

Provides information needed to apply utility-based reinforcement learning behaviors to agents. One row is filled out for each behavior as shown in Table 7.3.27.

Column/Field Name	Type	Description
name	String	The name of the UtilityBehavior
behavior	String	The name of the behavior declared in Table 7.3.25. Utility-based reinforcement learning will be applied to this behavior.
class	String	The class name of a UtilityBehavior defined in the rucg.mas.behavior.utility package. (See Note 1.)
normWeight	double	The contribution of the Subjective Norm to the Intention to perform the Behavior in the range [0.0, 1.0]. (See Note 2.)
attitudeWeight	double	The contribution of the Attitude Toward the Behavior to the Intention to perform the Behavior in the range [0.0, 1.0]. (See Note 2.)
controlWeight	double	The contribution of the Perceived Behavioral Control to the Intention to perform the Behavior in the range [0.0, 1.0]. (See Note 2.)
lambda	double	The discount rate for calculating the utility in the range [0.0, 1.0].
initialTemperature	double	The initial temperature for generating the Boltzmann distribution in the range (0.0, ∞). (See Note 3.)
minTemperature	double	The minimum temperature allowed in the range (0.0, ∞). (See Note 3.)

Table 7.3.27 UtilityBehavior Worksheet/Table

Note 1: The following UtilityBehavior classes are currently implemented:

a. SimpleUtilityBehavior
b. UtilityCommBehavior
c. UtilityInfrastructureTpBehavior

The package name "rucg.mas.behavior.utility" may be optionally prefixed to the class entry. Therefore, "UtilityInfrastructureTpBehavior" and "rucg.mas.behavior.utility.UtilityInfrastructureTpBehavior" are legitimate entries (without the quotes).

Note 2: The values of normWeight, attitudeWeight and controlWeight must sum to 1.0.

Note 3: initialTemperature must be greater than minTemperature.

7.3.28 UtilityIssues Worksheet/Table

Lists the issues evaluated by each UtilityBehavior. A row must be filled out for each issue considered by the UtilityBehavior as shown in Table 7.3.28.

Column/Field Name	Type	Description
utilityBehavior	String	The name of the UtilityBehavior defined in Table 7.3.27
issue	String	The name of the IssuePrototype defined in Table 7.3.5. This is the issue that will be evaluated by *utilityBehavior*.
weight	double	The weight that *issue* contributes in the range [0.0, 1.0]. The weights of all issues considered by *utilityBehavior* must sum to 1.0.
position	String	The name of the IssuePositionPrototype defined in Table 7.3.6. This is *issue's* position whose probability or likelihood will be evaluated by *utilityBehavior*.

Table 7.3.28 UtilityIssues Worksheet/Table

7.3.29 Method Worksheet/Table

A Method holds a set of method levels where each level determines one or more courses of action available to an agent. A method level provides a means to classify the condition of an agent. For example, an agent's condition may be classified to be one of five levels called "Very Positive", "Positive", "Neutral", "Negative", "Very Negative". These five levels would be grouped into one Method. This worksheet simply defines how the levels are grouped. The mappings from level to courses of action are entered in 7.3.30, "BehaviorMethodAction Worksheet/Table". One row must be filled out for each level as shown in Table 7.3.29.

Column/Field Name	Type	Description
name	String	The name of the Method
level	String	The name of the level
satisfactionThreshold	double	The satisfaction threshold associated with *level*

		in the range [0.0, 1.0].

Table 7.3.29 Method Worksheet/Table

The condition of an agent is determined by its satisfaction value calculated during the MetaCognition process; therefore, each level has a satisfaction threshold to determine which condition the agent will be classified under. The satisfaction value is a real number in the range [0.0, 1.0]. The larger the value, the more "satisfied" the agent is with the current state of affairs. The satisfaction threshold, therefore, must also be in the range [0.0, 1.0].

Example: If we use the five previously mentioned levels, we can assign the thresholds as follows:

Level	Threshold value
Very Negative	0.25
Negative	0.4
Neutral	0.6
Positive	0.75
Very Positive	1.0

SIM will interpret this to mean that if the agent's satisfaction value is less than or equal to 0.25, the agent is Very Negative; if greater than 0.25 and less than or equal to 0.4, the agent is Negative; if greater than 0.4 and less than or equal to 0.6, the agent is Neutral; if greater than 0.6 and less than or equal 0.75, the agent is Positive; and if greater than 0.75, the agent is Very Positive.

7.3.30 BehaviorMethodAction Worksheet/Table

Defines one or more courses of action (intent node states) that an agent can take given the method level it is currently classified to. One row must be filled out for each method level/ intent node state pair as shown in Table 7.3.30.

Column/Field Name	Type	Description
behavior	String	The name of the behavior declared in Table 7.3.25.
method	String	The name of the method declared in Table 7.3.29.
level	String	The name of the level declared in Table 7.3.29. The entries in this field and *method* must be consistent with the entries in the “name” and “level” fields in Table 7.3.29.
intentNodeState	String	The name of the state from the node representing the Intention to perform the behavior. The entries in this field and *behavior* must be consistent with the entries in the “behaviorName” and “intentNodeState” fields in Table 7.3.26.

Table 7.3.30 BehaviorMethodAction Worksheet/Table

7.3.31 AgentBehaviors Worksheet/Table

Declares the planned behaviors that each agent will simulate. It sets the initial state of the agent's behavior, the method the agent uses to select the action it will take, and determines how frequently the behaviors are carried out. Each row is filled out according to Table 7.3.31.

Column/Field Name	Type	Description
agent	String	The name of the agent defined in Table 7.3.20.
behaviorName	String	The name of the behavior declared in Table 7.3.25.
utilityBehavior	String	If applicable, the name of the UtilityBehavior defined in Table 7.3.27.
consumableType	String	If the "behaviorType" field in Table 7.3.25 is "INFRASTRUCTURE" for *behaviorName*, the name of the ConsumableType from Table 7.3.14 that will be restocked.
initialExecuteTime	String	Enter "NONE" or the distribution to generate the (simulation) time that this behavior will be executed for the first time. (See Notes 1 and 2.)
executeInterval	String	Enter "NONE" or the distribution to generate a waiting time before this behavior is repeated. (See Notes 1 and 2.)
stopBehaviorTime	String	Enter the distribution to generate the (simulation) time to stop this behavior. Enter "NONE" if this behavior should never be stopped. (See Notes 2 and 3.)
intentSelection	String	Enter "DRAW", "HIGHEST", or "THRESHOLD". (See Note 4.)
threshold	double	If intentSelection is "THRESHOLD", the threshold value in the range [0.0, 1.0].

Table 7.3.31 AgentBehaviors Worksheet/Table.

Note 1: Enter "NONE" if the "behaviorType" field in Table 7.3.25 is either "INFRASTRUCTURE" or "COMMUNICATE" for *behaviorName*; otherwise, enter the distribution according to Note 2.

Note 2: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 3: If a behavior is stopped it cannot be re-started until the start of the next replication.

Note 4: The agent can use one of three methods to choose an intention node state, i.e., choose the action it will perform:

a. DRAW – Perform a probability draw.
b. HIGHEST – Choose the state with the highest probability.
c. THRESHOLD – Declare a threshold value in the range [0.0, 1.0] and all states whose probability exceeds this value will be executed.

7.3.32 Location Worksheet/Table

Defines geographic areas within the AO. Locations are referenced by agents/groups, infrastructure, and actions. These areas are assumed to be polygons. One row of data is entered for each location as shown in Table 7.3.32.

Column/Field Name	Type	Description
name	String	The name of the location.
Level	String	Determines what level the Location is in the hierarchy of locations.
class	String	The class name of a Location defined in the rucg.mas.location package. (See Note 1.)
coordinate	String	The center coordinate of this location. Required only if *class* is "HexLocation" or "rucg.mas.location.HexLocation". The format depends upon the coordinate system. (See Note 2.) Ignored if class is neither "HexLocation" nor "rucg.mas.location.HexLocation".
numberVertices	int	The number of vertices this location owns. Enter a positive value only if there is a need to find a location given a coordinate; otherwise enter zero.
vertexCoordinate1, vertexCoordinate2, etc.	String	If *numberVertices* is positive, enter the first vertex coordinate under vertexCoordinate1, the second vertex coordinate under vertexCoordinate2, and so on. The format depends upon the coordinate system. (See Note 2.) Ignored if *numberVertices* is zero.

Table 7.3.32 Location Worksheet/Table.

Note 1: The following Location classes are currently implemented:

a. AreaLocation – A coarse representation of a geographic area in the AO.
b. HexLocation – A Location represented by a hexagon. All hexagons in a grid are assumed to be regular hexagons (i.e., all sides are equal in length and all internal angles are 120°).

The package name "rucg.mas.location" may be optionally prefixed to the class entry. Therefore, "AreaLocation" and "rucg.mas.location.AreaLocation" are legitimate entries (without the quotes).

Note 2: Locations may use one of the four coordinate systems listed below.

ARBITRARY_X_Y references an arbitrary x-y-Cartesian coordinate system. (It is primarily used for unit testing and is included in this document for completeness.)
GEODETIC coordinates reference latitude and longitude.
MILGRID (Military Grid Reference System) coordinates reference a zone number, three zone letters, and an easting and northing.
UTM (Universal Transverse Mercator) coordinates reference a zone number, the hemisphere (either north or south), and an easting and northing.

Locations may use a mix of GEODETIC, MILGRID, and UTM coordinates. MILGRID and UTM coordinates are automatically converted and stored internally as GEODETIC coordinates. ARBITRARY_X_Y coordinates cannot be mixed with the other three coordinates.

The format of the coordinate string is basically the coordinate system name followed by one or more parameters enclosed in parentheses. If the coordinate contains two or more parameters, the parameters are separated by commas.

ARBITRARY_X_Y contains two parameters. The first is the x-coordinate and the second is the y-coordinate. Example: ARBITRARY_X_Y(1,2).
GEODETIC contains two parameters. The first is the latitude in decimal degrees (in the range [-90.0, 90.0]) and the second is the longitude in decimal degrees (in the range [-180.0, 180.0]). Example: GEODETIC(-45.0, 60.0).
MILGRID contains one parameter. The parameter is an alpha-numeric string that starts with a zone number (1-60) followed by three zone letters, and ending with easting and northing (each up to five digits long). Examples: MILGRID(10SFF), MILGRID(10SFF04), MILGRID(10SFF0349), MILGRID(10SFF035496), MILGRID(10SFF03504968), MILGRID(10SFF0350649680).
UTM contains four parameters. The first is the zone number (1-60), the second is the hemisphere (either “N” or “S”), the third is the easting in meters and the fourth is the northing in meters. Example: UTM(10,N,555170,4163728).

7.3.33 LocationTree Worksheet/Table

Defines a hierarchy for locations where a lower level location represents a subdivision of the location that’s one level above it. This hierarchy is specified in terms of parent and child pairs. One row of data is entered for each parent/child pair as shown in Table 7.3.33.

Column/Field Name	Type	Description
parent	String	The name of the location defined in Table 7.3.32 that represents the parent in the hierarchy.
child	String	The name of the location defined in Table 7.3.32 that represents the child in the hierarchy.

Table 7.3.33 LocationTree Worksheet/Table.

7.3.34 LocationNeighbor Worksheet/Table

Defines the locations that are immediately adjacent to each other at a given hierarchal level. Neighboring locations are specified in terms of location pairs where one row of data is entered for each pair as described in Table 7.3.34. This table is optional and needs to be present and filled only if the "spatialMethod" field for the SimpleActionUmpire (Table 7.3.43) and SimpleInfrastructureUmpire (Table 7.3.54) is "COLLOCATION".

Column/Field Name	Type	Description
location1	String	The name of a location defined in Table 7.3.32.
location2	String	The name of a second location defined in Table 7.3.32 that is at the same hierarchal level as *location1* and is also directly adjacent to it.

Table 7.3.34 LocationNeighbor Worksheet/Table.

7.3.35 SimpleLocationUmpire Worksheet/Table

Defines the SimpleLocationUmpire. Only one row of data is required. The data is described in Table 7.3.35.

Column/Field Name	Type	Description
name	String	The name of the umpire.
assignLocationOnce	String	Enter "TRUE" if random assignment of Locations occurs once, or "FALSE" if random assignment occurs for each replication. (See Note, below.)
minXY	String	The minimum (southwestern-most) coordinate in the AO. The format depends upon the coordinate system. (See Note 2 under 7.3.32 "Location Worksheet/Table".) If the locations defined in Table 7.3.32 are hexagons, enter the coordinate of the southwestern-most hexagon.
maxXY	String	The minimum (southwestern-most) coordinate in the AO. The format depends upon the coordinate system. (See Note 2 under 7.3.32 "Location Worksheet/Table".) If the locations defined in Table 7.3.32 are hexagons, enter the coordinate of the northeastern-most hexagon.

Table 7.3.35 SimpleLocationUmpire Worksheet/Table.

Note: The SimpleLocationUmpire assigns an initial location to each agent at the start of each replication. If the agent's "initialLocation" field from Table 7.3.20 has a Location specified, the umpire will always assign that Location to the agent. If the agent's "initialLocation" is "ANY", however, the umpire will randomly assign a Location to the agent. Given that the umpire will randomly assign a Location to an agent, if the umpire's "assignLocationOnce" flag is "FALSE", the agent will be randomly assigned a Location at the start of each replication. If the flag is "TRUE', however, the agent will be randomly assigned a Location only once at the start of the first replication. The umpire subsequently reassigns the same Location at the start of each succeeding replication.

7.3.36 ActionType Worksheet/Table

Actions are categorized by their ActionType. An ActionType is used to classify a particular state of an intent node in a Bayesian network representing a planning behavior. (Compare this to “behaviorType” described in 7.3.25, “Behavior Worksheet/Table”.) Each row of data defines an ActionType as shown in Table 7.3.36.

Column/Field Name	Type	Description
Name	String	The name of the ActionType.
Class	String	The class name of an ActionType defined in the rucg.mas.action package. (See Note 1.)
commPriority	int	The priority to communicate about an action of this type. (See Note 2.)

Table 7.3.36 ActionType Worksheet/Table.

Note 1: The following ActionType classes are currently implemented:

a. CommunicationActionType – An action aimed at communicating the effects of another action to agents in the social network.
b. DamageActionType – An action aimed at damaging an infrastructure.
c. RepairActionType – An action aimed at repairing an infrastructure.
d. ResupplyActionType – An action taken by an agent to restock a consumable at an infrastructure.
e. KineticActionType – A kinetic action taken by an agent against another agent. Examples are insurgents attacking the coalition force agent, and the coalition force detaining/arresting members of the civilian populace.
f. NonKineticActionType – A non-kinetic action taken by an agent (usually to aid another agent). An example is the coalition force providing humanitarian aid to the civilian populace.
g. KLEActionType - An action taken by an agent to attend a Key Leader Engagement.
h. DoNothingActionType – No action taken.
i. ActivateInfraActionType – An action that allows an infrastructure to enter the simulation. This type of action should be used with DeactivateInfraActionType to simulate an infrastructure that is temporarily available to the civilian population, for example, a medical clinic (MEDCAP, DENTCAP or VETCAP). Actions of this type can only be triggered through scripted Actions. (See 7.3.37, “ScriptedAction Worksheet/Table”.)
j. DeactivateInfraActionType - An action taken to remove an infrastructure from the simulation. This type of action should be used with ActivateInfraActionType to simulate an infrastructure that is temporarily available to the civilian population. Actions of this type can only be triggered through scripted Actions. (See 7.3.37, “ScriptedAction Worksheet/Table”.)

The package name “rucg.mas.action” may be optionally prefixed to the class entry, and the ending “ActionType” may also be left off of the entry. Therefore, “Communicaton”, "CommunicationActionType", "rucg.mas.action.Communicaton", and "rucg.mas.action.CommunicationActionType" are legitimate entries (without the quotes).

Note 2: If an agent in the civilian population decides to communicate and that agent has more than one action to talk about, the agent chooses the action to talk about based on this field. A value of 1 represents the highest priority, 2 represents the second highest priority, and so on. SIM utilizes a min-priority queue to sort the actions from lowest "commPriority" value (highest priority) to highest "commPriority" value (lowest priority).

If a negative value or zero is entered, the agent will never communicate about this action.

If the "commPriority" column is missing, default values are applied based on the ActionType as follows:

ActionType	Default commPriority
KineticActionType	1
DamageActionType	1
NonKineticActionType	2
RepairActionType	3
ResupplyActionType	4
DoNothingActionType	-1
ActivateInfraActionType	-1
DeactivateInfraActionType	-1

7.3.37 ScriptedAction Worksheet/Table

Scripted actions are entered here. Scripted actions can be used to compensate for agent behaviors that have yet to be implemented, as well as be used to supplement existing behaviors. One row must be filled out for each action as shown in Table 7.3.37.

Column/Field Name	Type	Description
effectsIndex	int	A number used with the "index" column in the ScriptedEffects and ScriptedAttitudeEffects worksheets to link this action to the effects it has on an agent's or group's set of beliefs, values and interests. Enter -1 if this action has no effect.
initiator	String	The name of an agent defined in Table 7.3.20. This is the agent who initiates this action.
actionType	String	The name of an ActionType from Table 7.3.36.
target	String	The name of an agent defined in Table 7.3.20 if this action is targeted at a specific agent, or "ANY" if no agent is specifically targeted. (A blank will be interpreted as "ANY".) If this action is targeted at a specific infrastructure, enter the name of the infrastructure defined in

		Table 7.3.50.
keyWord	String	If *actionType* is the name of a DamageActionType, enter “LOW”, “MEDIUM” or “HIGH”, or enter the name of an infrastructure State defined in Table 7.3.49. (See Note 1.) If *actionType* is the name of a RepairActionType, enter the name of an infrastructure State or leave blank. (See Note 2.)
executeTimeClass	String	The distribution to generate the (simulation) time that this action will be executed for the first time. (See Note 3.)
repeat	int	The number of times to repeat this action.
repeatInterval	String	The distribution to generate a waiting time before this action is repeated. Ignored if *repeat* is zero or less. (See Note 3.)
location	String	If *target* is “ANY” or blank, enter the name of a Location defined in Table 7.3.32 if this action initially affects agents at this Location only; however, if the effects can be initially felt by agents anywhere in the AO, enter “ANY”. This column is ignored if *target* is the name of an agent or infrastructure.

Table 7.3.37 ScriptedAction Worksheet/Table.

Note 1: Scripted DamageActionTypes allow SIM to damage infrastructure in one of two ways. One way is to specify the intensity of the initiator’s attack by entering “LOW”, “MEDIUM” or “HIGH” as the key word. SIM will use this key word and the information in the Damage worksheet (Table 7.3.57) to determine the state of the infrastructure as a result of the attack. The second way to damage infrastructure is to specify the infrastructure’s new state as the key word. SIM will simply update the state of the infrastructure with this value at the time the scripted action is executed.

Note 2: Like scripted DamageActionTypes, scripted RepairActionTypes can be processed by SIM in two ways. If the key word is left blank, SIM uses the information in the Repair worksheet (Table 7.3.58) to determine the state of the infrastructure after repair and how long the repair will take. If the key word specifies the infrastructure’s new state, however, SIM will update the state of the infrastructure with this value at the time the scripted action is executed.

Note 3: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.38 ScriptedEffects Worksheet/Table

Defines the effects of a ScriptedAction on an agent’s or group’s set of beliefs, values, interests and positions on issues. Each issue (with its supporting beliefs, values and interests) is maintained in a Bayesian network and each effect is represented by a case file. One row must be filled out for each effect as shown in Table 7.3.38.

Column/Field Name	Type	Description
index	int	The index number of an action defined in Table 7.3.37. The number links this effect with the action.
beliefPrototype	String	The name of the BeliefPrototype defined in Table 7.3.3. This is the belief that is affected by the action referenced by *index*.
beliefPositionPrototype	String	The name of the BeliefPositionPrototype defined in Table 7.3.4. This is the position of *beliefPrototype* that is affected by the action referenced by *index*.
initiator	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that conducted the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that receives this effect.
receiverAttitude	String	Enter “NEGATIVE”, “NEUTRAL” or “POSITIVE”. This is *receiver*’s attitude towards *initiator* at the time the effect is received.
priorDistribution	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. The distribution should only generate values in the range [0.0, 1.0]. (See Note.)

Table 7.3.38 ScriptedEffects Worksheet/Table.

Note: The distribution is used to generate a probability. If the value generated by “priorDistribution” is less than zero (0.0), SIM sets the probability to 0.0; likewise, if the value generated by “priorDistribution” is greater than one (1.0), SIM sets the probability to 1.0. The distribution is specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses, e.g., Triangle(0.4, 0.7, 0.6).

7.3.39 ScriptedAttitudeEffects Worksheet/Table

Defines the effects of a ScriptedAction on an agent’s or group’s set of beliefs, values, interests and attitudes. Each attitude (with its supporting beliefs, values and interests) is maintained in a Bayesian network and each effect is represented by a case file. One row must be filled out for each effect as shown in Table 7.3.39.

Column/Field Name	Type	Description
index	Int	The index number of an action defined in Table 7.3.37. The number links this effect with the

		action.
beliefPrototype	String	The name of the BeliefPrototype defined in Table 7.3.3. This is the belief that is affected by the action referenced by *index*.
beliefPositionPrototype	String	The name of the BeliefPositionPrototype defined in Table 7.3.4. This is the position of *beliefPrototype* that is affected by the action referenced by *index*.
initiator	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that conducted the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that receives this effect.
priorDistribution	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. The distribution should only generate values in the range [0.0, 1.0]. (See Note.)

Table 7.3.39 ScriptedAttitudeEffects Worksheet/Table.

Note: The distribution is used to generate a probability. If the value generated by "priorDistribution" is less than zero (0.0), SIM sets the probability to 0.0; likewise, if the value generated by "priorDistribution" is greater than one (1.0), SIM sets the probability to 1.0. The distribution is specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses, e.g., Triangle(0.4, 0.7, 0.6).

7.3.40 BehaviorAction Worksheet/Table

Defines external operations initiated by an agent or group based on a planned behavior. One row must be filled out for each action as shown in Table 7.3.40.

The result of a BehaviorAction (either success or failure) may have an effect on an entity's set of beliefs, values and interests that, in turn, affects that entity's positions on issues and attitudes. The effects on beliefs, values and interests that affect positions on issues are entered in the IssueActonEffects worksheet described in 7.3.41, "IssueActionEffects Worksheet/Table". The effects on beliefs, values and interests that affect attitudes are entered in the AttitudeActionEffects worksheet described in 7.3.42, "AttitudeActionEffects Worksheet/Table".

Column/Field Name	Type	Description
index	int	A number used to identify this action. This number is used with the "index" column in the IssueActionEffects and AttitudeActionEffects worksheets to link the action with its effect(s).
behaviorName	String	The name of the behavior declared in Table 7.3.25.

intentNodeState	String	The name of the state from the node representing the Intention to perform the behavior. The entries in this field and *behaviorName* must be consistent with the entries in the "behaviorName" and "intentNodeState" fields in Table 7.3.26.
actionType	String	The name of an ActionType from Table 7.3.36. This is the ActionType that best characterizes the action associated with *intentNodeState*.

Table 7.3.40 BehaviorAction Worksheet/Table.

7.3.41 IssueActionEffects Worksheet/Table

Defines the effects of a BehaviorAction on an agent's or group's set of beliefs, values, and interests. Each effect is represented by a draw from a distribution. One row must be filled out for each effect as shown in Table 7.3.41.

Column/Field Name	Type	Description
Index	int	The index number of an action defined in Table 7.3.40. The number links this effect with the action.
beliefPrototype	String	The name of the BeliefPrototype defined in Table 7.3.3. This is the BeliefPrototype that is affected by the action referenced by *index*.
beliefPositionPrototype	String	The name of the BeliefPositionPrototype defined in Table 7.3.4. This is the position of *beliefPrototype* that is affected by the action referenced by *index*.
initiator	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that conducted the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that receives this effect.
consumableType	String	Match *index* from this table with the index number in Table 7.3.40 and obtain the associated "behaviorName" from Table 7.3.40. If the "behaviorType" field in Table 7.3.25 is "INFRASTRUCTURE" for "behaviorName", enter the name of the ConsumableType that "behaviorName" is used to restock. Ignored if "behaviorType" is not "INFRASTRUCTURE".
providerAssociation	String	Match *index* from this table with the index number in Table 7.3.40 and obtain the associated "behaviorName" from Table 7.3.40. If the "behaviorType" field in Table 7.3.25 is "INFRASTRUCTURE" for "behaviorName",

		enter the name of an AgentPrototype defined in Table 7.3.15 whose “isExternal” field is “TRUE”. Ignored if “behaviorType” is not “INFRASTRUCTURE”.
outcome	String	Enter either “SUCCESS” or “FAIL”. This is the outcome of the action referenced by *index*.
receiverAttitude	String	Enter “NEGATIVE”, “NEUTRAL” or “POSITIVE”. This is *receiver*’s attitude towards *providerAssociation* at the time this effect is received, if applicable; otherwise this is *receiver*’s attitude towards *initiator*.
priorDistribution	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. The distribution should only generate values in the range [0.0, 1.0]. (See Note.)

Table 7.3.41 IssueActionEffects Worksheet/Table.

Note: The distribution is used to generate a probability. If the value generated by “priorDistribution” is less than zero (0.0), SIM sets the probability to 0.0; likewise, if the value generated by “priorDistribution” is greater than one (1.0), SIM sets the probability to 1.0. The distribution is specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses, e.g., Triangle(0.4, 0.7, 0.6) .

7.3.42 AttitudeActionEffects Worksheet/Table

Defines the effects of a BehaviorAction on an agent’s or group’s set of beliefs, values, and interests. Each effect is represented by a draw from a distribution. One row must be filled out for each effect as shown in Table 7.3.42.

Column/Field Name	Type	Description
Index	int	The index number of an action defined in Table 7.3.40. The number links this effect with the action.
beliefPrototype	String	The name of the BeliefPrototype defined in Table 7.3.3. This is the BeliefPrototype that is affected by the action referenced by *index*.
beliefPositionPrototype	String	The name of the BeliefPositionPrototype defined in Table 7.3.4. This is the position of *beliefPrototype* that is affected by the action referenced by *index*.
initiator	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that conducted the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.15. This is the AgentPrototype that receives this effect.

consumableType	String	Match *index* from this table with the index number in Table 7.3.40 and obtain the associated “behaviorName” from Table 7.3.40. If the “behaviorType” field in Table 7.3.25 is “INFRASTRUCTURE” for “behaviorName”, enter the name of the ConsumableType that “behaviorName” is used to restock. Ignored if “behaviorType” is not “INFRASTRUCTURE”.
providerAssociation	String	Match *index* from this table with the index number in Table 7.3.40 and obtain the associated “behaviorName” from Table 7.3.40. If the “behaviorType” field in Table 7.3.25 is “INFRASTRUCTURE” for “behaviorName”, enter the name of an AgentPrototype defined in Table 7.3.15 whose “isExternal” field is “TRUE”. Ignored if “behaviorType” is not “INFRASTRUCTURE”.
outcome	String	Enter either “SUCCESS” or “FAIL”. This is the outcome of the action referenced by *index*.
priorDistribution	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. The distribution should only generate values in the range [0.0, 1.0]. (See Note.)

Table 7.3.42 AttitudeActionEffects Worksheet/Table.

Note: The distribution is used to generate a probability. If the value generated by “priorDistribution” is less than zero (0.0), SIM sets the probability to 0.0; likewise, if the value generated by “priorDistribution” is greater than one (1.0), SIM sets the probability to 1.0. The distribution is specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses, e.g., Triangle(0.4, 0.7, 0.6) .

7.3.43 SimpleActionUmpire Worksheet/Table

Provides rules for how the SimpleActionUmpire operates. Only one row of data is required. The data is described in Table 7.3.43.

Column/Field Name	Type	Description
name	String	The name of the umpire.
recipientsNoTarget	int	The number of agents to choose at random who will receive the effects of an action if that action does not specify a target.
recepientsInfra	int	The number of agents to choose at random who will receive the effects of an action if that action specifies an infrastructure target.
doNotPassInterval	double	A period of time during which an agent will only pass an action once to other agents in its social

		network. (See Note 1.)
sociabilityMethod	String	Enter “K_NEAREST_NEIGHBOR” or “K_TRIM_THRESHOLD”. (See Note 2.)
sociabilityClass	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. The distribution must be consistent with the *sociabilityMethod*. (See Note 2.)
commOrder	String	Enter “DESCENDING_ORDER” or “RANDOM_ORDER”. (See Note 3.)
spatialMethod	String	Enter “COLLOCATION” or PROXIMITY. (See Note 4.)
proximityRadius	String	If *spatialMethod* is PROXIMITY, the maximum distance that two agents can be apart to have an opportunity to communicate. (See Note 5.) Ignored if *spatialMethod* is COLLOCATION.
commTimeClass	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the time it takes for an agent to communicate the effects of an action to another agent.

Table 7.3.43 SimpleActionUmpire Worksheet/Table.

Note 1: This column addresses a situation where an agent may receive information about the same action from other agents in its social network within a short time interval. Under this situation, when the agent receives the action for the first time, the agent will attempt to pass the action to the other agents in the network. If the agent receives the same action from another member of the network during this time interval, the agent will process the action, but will not pass the action to the other agents in the network. The time interval is defined by this column.

Note 2: This column provides SIM the capability to consider sociability to determine who an agent will communicate with. Two methods called K_NEAREST_NEIGHBOR and K_TRIM_THRESHOLD have been implemented.

The k nearest neighbor is a method where an agent determines the number of other agents k it will communicate with based on a draw from a distribution. The number k is the k agents closest in social space.

The k trim threshold is a method where an agent determines who to communicate with based on the maximum social distance k within which the agent will consider communicating. Agent i will only consider communicating with agent j if the social distance between them is less than k. The distance k, also called the “trim”, is drawn from a distribution. The acceptable range for k is approximately [0.7, 0.99] depending upon the number of agents in the scenario.

For both methods the “sociabilityClass” column is used to enter the appropriate distribution for generating the value of k. It is up to the analyst to insure that the distribution entered in this column generates k values that are consistent with the method entered in the “sociabilityMethod” column.

Note 3: This column is used to determine the order in which an agent will communicate with other agents after that agent has determined who to communicate with (using either K_NEAREST_NEIGHBOR or K_TRIM_THRESHOLD discussed in Note 2, above.) The order may be either DESCENDING_ORDER or RANDOM_ORDER.

When the agent uses descending order, the order of communication is in descending order of similarity based on social distance.

When an agent uses random order, the order of communication is in a random order.

Note 4: This column provides SIM the capability to consider spatial distance between agents as a criterion to communicate with each other. Two methods are available called COLLOCATION and PROXIMITY.

Collocation is a method where communication between two agents is conditioned on whether the agents occupy the same Location (defined in Table 7.3.32) or directly adjacent Locations (as defined in Table 7.3.34).

Proximity is a method where an agent will consider communicating with other agents that are within a specified distance radius.

Note 5: The distance is measured in kilometers if Locations use GEODETIC, MILGRID, or UTM coordinates as specified by the “coordinate” field in Table 7.3.32. If Locations use ARBITRARY_X_Y coordinates, however, the distance is measured in an arbitrary unit consistent with that coordinate system.

7.3.44 SimpleHomophilyNetworkUmpire Worksheet/Table

Defines the SimpleHomophilyNetworkUmpire. Only one row of data is required. The data is described in Table 7.3.44.

Column/Field Name	Type	Description
name	String	The name of the umpire.
linkWeightUpdateInterval	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the time between link weight updates for every agent pair in the civilian population.

Table 7.3.44 SimpleHomophilyNetworkUmpire Worksheet/Table.

7.3.45 RandomConsumptionLogic Worksheet/Table

Defines the parameters for random consumption of a ConsumableType. Each row defines the consumption logic used for a ConsumableType defined in Table 7.3.14. The data is described in Table 7.3.45.

Column/Field Name	Type	Description
name	String	The name of the consumption logic.
start	String	The distribution for the time of the first consumption event. (See Note.)
amount	String	The distribution for the amount of the ConsumableType consumed at each consumption event. (See Note.)
nominalInterval	String	The distribution for the nominal time between consumption events. (See Note.)
timeVariability	String	The distribution of the time variability applied to the nominal time interval.
consumableType	String	The name of a ConsumableType defined in Table 7.3.14 that is consumed at each consumption event.

Table 7.3.45 RandomConsumptionLogic Worksheet/Table.

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.46 AgentConsumables Worksheet/Table

Specifies the types of consumables consumed by agents. One row must be filled out for each ConsumableType consumed by an agent. The data is described in Table 7.3.46.

Column/Field Name	Type	Description
agent	String	The name of an agent defined in Table 7.3.20.
consumableType	String	The name of a ConsumableType defined in Table 7.3.14.
initialQuantity	double	The initial quantity of *consumableType* that *agent* holds.
consumptionLogic	String	The name of the ConsumptionLogic defined in Table 7.3.45 that will be used to determine how *agent* consumes *consumableType*.
maxCapacity	double	The maximum amount of *consumableType* that *agent* can hold.
maxRefill	double	The maximum amount of *consumableType* that *agent* can obtain per visit to an infrastructure.
restock	double	The *agent* creates a requirement when the current amount of stock drops below this quantity expressed as a fraction of *maxCapacity*.

Table 7.3.46 AgentConsumables Worksheet/Table.

7.3.47 InfrastructureType Worksheet/Table

Infrastructures are classified by InfrastructureType. Each row of data defines an InfrastructureType as shown in Table 7.3.47.

Column/Field Name	Type	Description
name	String	The name of the type of infrastructure.
consumableType	String	The name of a ConsumableType defined in Table 7.3.14. This is the good or service provided by infrastructures of this type.
isUtility	String	Enter "TRUE" if this type of infrastructure represents a utility AND there is a need to override the PROXIMITY/COLLOCATION "spatialMethod" of the SimpleInfrastructureUmpire defined in Table 7.3.54; otherwise, enter "FALSE". (See Note 1.)
isMobile	String	Indicates whether an infrastructure of this type can be moved during the simulation. Enter "TRUE" if this is the case; otherwise, enter "FALSE". (See Note 2.)

Table 7.3.47 InfrastructureType Worksheet/Table.

Note 1: If set to "TRUE", the UtilityServiceArea must be filled out for every infrastructure in Table 7.3.50 that is specified to be of this InfrastructureType. Only agents in the defined service area will receive the service provided by the infrastructure. These agents also will not be able to seek an alternate provider of that service.

Note 2: An InfrastructureType cannot have both "isUtility" and "isMobile" set to "TRUE".

7.3.48 AgentProtoInfraTypeData Worksheet/Table

Provides renege time and balk information by AgentProtoype and InfrastructureType. One row is filled out for each AgentProtoype-InfrastructureType pair as shown in Table 7.3.48.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.15 whose "isExternal" field is "FALSE".
infrastructureType	String	The name of the InfrastructureType defined in Table 7.3.47.
renegeTime	String	The distribution for generating renege times when an agent of type *agentPrototype* enters a queue of an infrastructure of type *infrastructureType*. (See Note 1.)
balkThreshold	int	The acceptable limit of agents in the queue

		before an agent of type *agentPrototype* balks at an infrastructure of type *infrastructureType*. The value may be positive, zero, or negative. (See Note 2.)
proximityRadius	double	If the "spatialMethod" field of the SimpleInfrastructureUmpire defined in Table 7.3.54 is "PROXIMITY", enter the maximum distance that an agent of type *agentPrototype* is willing to seek an infrastructure of type *infrastructureType*. (See Note 3.) Ignored if the "spatialMethod" field of the SimpleInfrastructureUmpire is "COLLOCATION".

Table 7.3.48 AgentProtoInfraTypeData Worksheet/Table.

Note 1: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 2: Given a threshold t > 0, the agent will enter the queue if there are t or fewer agents already waiting in the queue, and the agent will balk if there are t + 1 agents (or more) already in the queue. If t = 0, the agent will only enter the queue if the queue is empty; otherwise, the agent balks. If t is negative, the agent will always enter the queue as long as the number of agents already in the queue has not reached the queue's capacity; if the queue is already at capacity, the agent will balk.

Note 3: The distance is measured in kilometers if Locations use GEODETIC, MILGRID, or UTM coordinates as specified by the "coordinate" field in Table 7.3.32. If Locations use ARBITRARY_X_Y coordinates, however, the distance is measured in an arbitrary unit consistent with that coordinate system.

7.3.49 InfrastructureState Worksheet/Table

Defines the states of an infrastructure. The states affect the operational parameters of the infrastructure specified in 7.3.51 "InfrastructureOperation Worksheet/Table". Each row of data defines an InfrastructureState as shown in Table 7.3.49.

Column/Field Name	Type	Description
name	String	The name of the state.
inoperable	String	Indicates whether an infrastructure in this state is inoperable. If "TRUE", the infrastructure cannot operate in this state until it is repaired to a state that has the "inoperable" flag set to "FALSE"

Table 7.3.49 InfrastructureState Worksheet/Table.

Example: Suppose an infrastructure can be defined to be in one of the four following states:

- Normal Baseline – the infrastructure is operating normally.
- Damaged Operable – damage to the infrastructure that causes a degradation of operational characteristics (specified in Table 7.3.51) compared to the Normal Baseline state.
- Damaged Non-Operable – damage to the infrastructure that causes it to be unable to serve any customers.
- Renovated or Retrofitted – the infrastructure's operational characteristics are improved over the Normal Baseline state.

The InfrastructureState worksheet should be filled out as follows:

name	inoperable
Baseline Normal	FALSE
Damaged-Operable	FALSE
Damaged-Inoperable	TRUE
Renovated	FALSE

InfrastrcutreState Worksheet Example.

7.3.50 InfrastructureServer Worksheet/Table

Defines infrastructure that agents visit to restock goods or receive services (i.e., ConsumableTypes). This class of infrastructure is represented by a multi-server queue with reneging and balking. One row of data is entered for each infrastructure as shown in Table 7.3.50.

Column/Field Name	Type	Description
name	String	The name of the infrastructure.
type	String	The name of the InfrastructureType defined in Table 7.3.47.
initialOpenTime	String	The distribution for the time of the first open event (i.e., "opens for business" for the first time). (See Note 1.) If this infrastructure is to enter the simulation by a scripted Action, however, enter "SCRIPTED", instead.
serviceInQueue	String	Indicates whether agents in the queue will be served when the infrastructure closes. If "TRUE", these agents will be served. If "FALSE", however, these agents will be immediately scheduled to depart the infrastructure.
initialLocation	String	The name of the Location defined in Table 7.3.32 where this infrastructure will be located at time 0. If *initialOpenTime* is "SCRIPTED", however, enter "SCRIPTED", instead.
initialOwner	String	The name of an agent defined in Table 7.3.20 that initially has overall responsibility for operating and maintaining this infrastructure.

initialState	String	Either the name of a State defined in Table 7.3.49, “RANDOM” or “SCRIPTED”. This is the state of the infrastructure at the start of each replication. (See Note 2.)
initialQuantity	double	The initial quantity of the good or service that this infrastructure provides.
association	String	The name of an AgentPrototype defined in Table 7.3.15 that is associated with this infrastructure. (See Note 3.)

Table 7.3.50 InfrastructureServer Worksheet/Table.

Note 1: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 2: If “RANDOM” is entered, the infrastructure is randomly assigned one of the states defined in Table 7.3.49 at the start of each replication. Enter “SCRIPTED” only if *initialOpenTime* is “SCRIPTED”.

Note 3: The “isExternal” field of the AgentPrototype (see Table 7.3.15) should be “TRUE”.

7.3.51 InfrastructureOperation Worksheet/Table

Provides operational characteristics for infrastructures based on their state. One row must be filled out for each infrastructure/state combination. The data is described in Table 7.3.51.

Column/Field Name	Type	Description
infrastructureName	String	The name of an InfrastructureServer defined in Table 7.3.50.
infrastructureState	String	The name of a State defined in Table 7.3.49.
adminTime	String	The distribution for generating any additional administrative or setup time the infrastructure needs to serve an agent. (See Note 1.)
openTime	String	The distribution for generating the time this infrastructure remains open. (See Note 1.)
closeTime	String	The distribution for generating the time this infrastructure remains closed. (See Note 1.)
numberServers	int	The number of servers.
queueCapacity	int	The maximum number of agents that the queue can hold.
maxRefill	double	The maximum amount of the good or

		service that a server can transfer to an agent per visit.
transferRate	String	The distribution for generating the amount of the good or service that a server can transfer to an agent per unit time. (See Note 1.)
consumableCapacity	double	The maximum amount of the good or service that the infrastructure can hold.
desired	double	The minimum amount that the infrastructure will request expressed as a fraction of *consumableCapacity*.
restock	double	The infrastructure creates a requirement when the current amount of stock drops below this quantity expressed as a fraction of *consumableCapacity*.
restockInterval	String	The distribution for generating the time it takes to restock. (See Note 1.)

Table 7.3.51 InfrastructureOperation Worksheet/Table.

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Example: Using the four states from the example in 7.3.49 “InfrastructureState Worksheet/Table”, suppose there is an infrastructure named Infra1. The InfrastructureOperation worksheet will have four rows filled out for Infra1, one for each state, as illustrated below.

infrastructureName	infrastructureState	adminTime	openTime	closeTime	numberServers
Infra1	Baseline-Normal	Exponential(0.025)	Normal(0.5, 0.001)	Normal(0.5, 0.001)	3
Infra1	Damaged-Operable	Exponential(0.0275)	Normal(0.25, 0.001)	Normal(0.75, 0.001)	2
Infra1	Renovated	Constant(0.0)	Normal(0.75, 0.001)	Normal(0.25, 0.001	4
Infra1	Damaged-Inoperable	Constant(0.0)	Constant(0.0)	Constant(365.0)	0

queueCapacity	maxRefill	transferRate	consumableCapacity	desired	restock	restockInterval
10	50.00	Uniform(4900, 5100)	1.50E+07	0.9	0.5	Normal(5.0, 0.001)
9	25.00	Uniform (3500, 4000)	1.00E+07	0.9	0.5	Normal(5.0, 0.001)
12	55.00	Uniform (6000, 6100)	1.60E+07	0.9	0.5	Normal(5.0, 0.001)
0	0.00	Constant(0)	0.00E+00	0	0	Constant(0.0)

InfrastructureOperation Worksheet Example.

7.3.52 UtilityServiceArea Worksheet/Table

This worksheet should only be filled out for those InfrastructureServers whose type has their “isUtility” field set to “TRUE” in Table 7.3.47. It specifies the Locations that define the area served by these InfrastructureServers. One row must be filled out for each Location in the service area. The data is described in Table 7.3.52.

Column/Field Name	Type	Description
infrastructureName	String	The name of an Infrastructure defined in Table 7.3.50.
location	String	The name of a Location defined in Table 7.3.32.

Table 7.3.52 UtilityServiceArea Worksheet/Table

7.3.53 AgentInfrastructureData Worksheet/Table

Specifies delay and cost information between an agent and an InfrastructureServer. One row must be filled out for each applicable agent/infrastructure pair. The data is described in Table 7.3.53.

Column/Field Name	Type	Description
agent	String	The name of an agent defined in Table 7.3.20.
infrastructureName	String	The name of an Infrastructure defined in Table 7.3.50.
delayClass	String	The distribution to generate the time it takes the agent to travel to or from the infrastructure. (1 and 2.)
cost	double	The overall cost that the agent incurs to restock from the infrastructure. (See Note 3.)

Table 7.3.53 AgentInfrastructureData Worksheet/Table.

Note 1: Delay distributions should be entered if the “spatialMethod” field of the SimpleInfrastructureUmpire defined in Table 7.3.54 is “COLLOCATION”. If the “spatialMethod” field is “PROXIMITY”, however, the “moveRate” field in table 7.3.15 should be used instead and “NA” should be entered in the “delayClass” field.

Note 2: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 3: The cost is only used by the SimpleInfrastructureUmpire to choose between two Infrastructures that provide the same ConsumableTypes.

7.3.54 SimpleInfrastructureUmpire Worksheet/Table

Defines the SimpleInfrastructureUmpire. Only one row of data is required. The data is described in Table 7.3.54.

Column/Field Name	Type	Description

name	String	The name of the umpire.
spatialMethod	String	Enter “COLLOCATION” or PROXIMITY. (See Note.)

Table 7.3.54 SimpleInfrastructureUmpire Worksheet/Table.

Note: This column provides SIM the capability to consider spatial distance between an agent and infrastructure to help the umpire determine which infrastructure the agent can visit. Two methods are available called COLLOCATION and PROXIMITY.

Collocation is a method where an infrastructure is considered only if the infrastructure and agent occupy the same Location (defined in Table 7.3.32) or directly adjacent Locations (as defined in Table 7.3.34). Collocation should be used if all agents and infrastructure are placed on AreaLocations. (See 7.3.32, “Location Worksheet/Table”.)

Proximity is a method where an infrastructure is considered only if the infrastructure and agent are within a specified distance radius. Proximity should be used if all agents and infrastructure are placed on HexLocations. (See 7.3.32, “Location Worksheet/Table”.)

7.3.55 SimpleDamageUmpire Worksheet/Table

Defines the SimpleDamageUmpires. An umpire is defined for each InfrastructureType, one row per umpire, as shown in Table 7.3.55.

Column/Field Name	Type	Description
name	String	The name of the umpire.
infrastructureType	String	The name of the InfrastructureType defined in Table 7.3.47 that this umpire assesses.

Table 7.3.55 SimpleDamageUmpire Worksheet/Table

7.3.56 RepairRule Worksheet/Table

Provides information to the DamageUmpire for prioritizing infrastructure needing repair or renovation based on the current state of the infrastructure. The worksheet also specifies the target state that repairing or renovating must attain. One row of data is entered for each DamageUmpire/infrastructure state combination as shown in Table 7.3.56.

Column/Field Name	Type	Description
damageUmpire	String	The name of the DamageUmpire defined in Table 7.3.55.
fromInfrastructureState	String	The name of a State defined in Table 7.3.49 and different from *toInfrastructureState*. This is the state of the infrastructure before repairs begin.
repairPriority	int	Indicates the priority for repair of an infrastructure in *fromInfrastructureState*. The higher the value, the higher the priority for repair. A negative value indicates that an infrastructure in *fromInfrastructureState* should

		not be considered for repair.
toInfrastructureState	String	The name of a State defined in Table 7.3.49 and different from *fromInfrastructureState*. This is the desired state of the infrastructure after repairs have been completed, given that the infrastructure is currently in *fromInfrastructureState*. Enter "NA" if *priority* is negative.

Table 7.3.56 RepairRule Worksheet/Table.

Example: Using the four states from the example in 7.3.49 "InfrastructureState Worksheet/Table", suppose there is a DamageUmpire named DamageUmpire1. Also, suppose for the type of infrastructure handled by DamageUmpire1 that infrastructure at the Damaged-Operable state are given the highest priority to repair, infrastructure at the Damaged-Inoperable state are given a lower priority to repair, and infrastructure at the Baseline-Normal state are given the lowest priority to repair (renovate or retrofit). There is no priority given to infrastructure at the Renovated state since, by definition, it is the state that has the best operational characteristics among the four states.

Finally, suppose the desired end-state for either a Damaged-Operable or Damaged-Inoperable infrastructure is Baseline-Normal and the desired end-state of a Baseline-Normal infrastructure is Renovated.

All of the information for DamageUmpire1 above can be entered in four rows in the RepairRule worksheet as follows:

damageUmpire	fromInfrastructureState	repairPriority	toInfrastructureState
DamageUmpire1	Baseline-Normal	1	Renovated
DamageUmpire1	Damaged-Operable	3	Baseline-Normal
DamageUmpire1	Damaged-Inoperable	2	Baseline-Normal
DamageUmpire1	Renovated	-1	NA

RepairRule Worksheet Example.

7.3.57 Damage Worksheet/Table

Provides the probability of damage on infrastructure based on the type of infrastructure, the prototype of the attacker, the intensity of the attack, the state of the infrastructure before the attack, and the state of the infrastructure after the attack. One row of data is entered for each infrastructure/attacker/intensity/state combination as shown in Table 7.3.57.

Column/Field Name	Type	Description
infrastructureType	String	The name of the InfrastructureType defined in Table 7.3.47.
attackerPrototype	String	The name of an AgentPrototype defined in Table 7.3.15.
intensity	String	Enter "LOW", "MEDIUM", or "HIGH". This is the intensity of the attack.

fromInfrastructureState	String	The name of a State defined in Table 7.3.49. This is the state of the infrastructure before the attack. (See Note.)
toInfrastructureState	String	The name of a State defined in Table 7.3.49. This is the state of the infrastructure after the attack. (See Note.)
probability	double	The probability that an attacker of type *attackerPrototype* attacking an infrastructure of type *infrastructureType* with intensity *intensity* will change the infrastructure's state from *fromInfrastructureState* to *toInfrastructureState*. (See Note.)

Table 7.3.57 Damage Worksheet/Table.

Note: If fromInfrastructureState and toInfrastructureState are the same, the probability is the probability that the attack does not cause any (extra) damage.

Each set of probabilities grouped by infrastructureType/attackerPrototype/intensity/fromInfrastructureState must sum to 1.0. This constraint is illustrated in the following example.

Example: Using the four states from the example in 7.3.49 "InfrastructureState Worksheet/Table", suppose there is an infrastructure type named InfraType1 and an attacker prototype named AttackerType1. For illustrative purposes consider the MEDIUM attack intensity. Since there are four states defined, there are four possible outcomes (possible change of state) if a MEDIUM intensity attack occurs when the infrastructure's state before the attack is Renovated: Renovated (no damage), Baseline-Normal (damage), Damaged-Operable (more damage) and Damaged-Inoperable (most damage). Similarly, there are three outcomes if the before-attack state is Baseline-Normal: Baseline-Normal (no damage), Damaged-Operable (damage) and Damaged-Inoperable (most damage). There are two outcomes if the before-attack state is Damaged-Operable: Damaged-Operable (no additional damage) and Damaged-Inoperable (more damage). Finally, there can be only one outcome if the before-attack state is Damaged-Inoperable: Damaged-Inoperable (no additional damage). The Damage worksheet will have up to 10 rows filled out that reflect the outcomes of the InfraType1/AttackerType1/MEDIUM combination as illustrated below (row numbers provided for reference).

	infrastructureType	attackerPrototype	intensity	fromInfrastructureState	toInfrastructureState	probability
1	InfraType1	AttackerType1	MEDIUM	Baseline-Normal	Baseline-Normal	0.0
2	InfraType1	AttackerType1	MEDIUM	Baseline-Normal	Damaged-Operable	0.99
3	InfraType1	AttackerType1	MEDIUM	Baseline-Normal	Damaged-Inoperable	0.01
4	InfraType1	AttackerType1	MEDIUM	Damaged-Operable	Damaged-Operable	0.4
5	InfraType1	AttackerType1	MEDIUM	Damaged-Operable	Damaged-Inoperable	0.6
6	InfraType1	AttackerType1	MEDIUM	Damaged-Inoperable	Damaged-Inoperable	1
7	InfraType1	AttackerType1	MEDIUM	Renovated	Renovated	0.0
8	InfraType1	AttackerType1	MEDIUM	Renovated	Baseline-Normal	0.0

9	InfraType1	AttackerType1	MEDIUM	Renovated	Damaged-Operable	0.89
10	InfraType1	AttackerType1	MEDIUM	Renovated	Damaged-Inoperable	0.11

Damage Worksheet Example.

Note that the probabilities in rows 1 through 3, rows 4 and 5, and rows 7 through 10 each sum to 1.0. Optionally, rows 1, 7 and 8 may be left out of the worksheet. In a similar manner, the worksheet may have up to 10 rows filled out each for the InfraType1/AttackerType1/LOW and InfraType1/AttackerType1/HIGH combinations.

7.3.58 Repair Worksheet/Table

Contains the repair time information for infrastructure based on the type of infrastructure, the prototype of the repairer, the state of the infrastructure before repairs begin, and the desired state of the infrastructure after repairs have been completed. This worksheet is not limited to repairs for damaged infrastructures. It can be used to enter times to renovate or retrofit undamaged infrastructures, as well. One row of data is entered for each infrastructure/repairer/state combination as shown in Table 7.3.58.

Column/Field Name	Type	Description
infrastructureType	String	The name of the InfrastructureType defined in Table 7.3.47.
repairerPrototype	String	The name of an AgentPrototype defined in Table 7.3.15.
fromInfrastructureState	String	The name of a State defined in Table 7.3.49 and different from *toInfrastructureState*. This is the state of the infrastructure before repairs begin.
toInfrastructureState	String	The name of a State defined in Table 7.3.49 and different from *fromInfrastructureState*. This is the desired state of the infrastructure after repairs have been completed.
repairTime	String	The distribution for generating the time it takes to get the infrastructure from *fromInfrastructureState* to *toInfrastructureState*. (See Note.)

Table 7.3.58 Repair Worksheet/Table.

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Example: Using the four states from the example in 7.3.49 “InfrastructureState Worksheet/Table”, suppose there is an infrastructure type named InfraType1 and a repairer prototype named RepairerType1. From the example in 7.3.56 “RepairRule Worksheet/Table”, there is a DamageUmpire named DamageUmpire1. Suppose this umpire handles infrastructures of type InfraType1. In the same example it was declared that infrastructures of the type handled by DamageUmpire1 are to be repaired to the Baseline-Normal state if they are currently in either the Damaged-Operable or Damaged-

Inoperable state, but they are to be renovated or retrofitted to the Renovated state if they are currently in the Baseline-Normal state. For a repairer of type RepairerType1 that may repair an infrastructure of type InfraType1, three rows of the Repair worksheet will be filled out as illustrated below.

infrastructureType	repairerPrototype	fromInfrastructureState	toInfrastructureState	repairTime
InfraType1	RepairerType1	Baseline-Normal	Renovated	Triangle(24, 26, 25)
InfraType1	RepairerType1	Damaged-Operable	Baseline-Normal	Normal(15, 0.1)
InfraType1	RepairerType1	Damaged-Inoperable	Baseline-Normal	Uniform(27, 28)

Repair Worksheet Example.

7.3.59 KLEParameters

Specifies general parameters effecting key leader engagements and dynamic social networks.

Column/Field Name	Type	Description
locationLevel	String	Determines what level of Locations has to match for key leader replacement.
probAffinityChangesRationally	double	The probability that the affinity between 2 participants changes based on the results of a key leader engagement
randomWalkProb	double	The probability that the affinity between 2 participants changes randomly after a key leader engagement
defaultAffinityLevel	String	The assumed initial affinity level from one entity to another if the relationship is not specified in the AffinityNetwork table
engagementUmpireFile	String	The name of a file that contains the XML fragment that defines the KeyLeaderEngagementUmpire. The format of the file is documented in paragraph 7.4 below.
replacementDelay	String	The distribution for the time it takes to replace a removed of killed key leader. (See note for format)
comintProbability	double	The probability that the occurrence of a behavior will be reported using COMINT.
homophilyWeight	Double	How much weight the dynamic network umpire puts on the social distance between 2 entities when deciding who should form relationships. Between 0 and 1. The sum of homophilyWeight and roleWeight should be 1.0
roleWeight	Double	How much to weight the dynamic network umpire puts on the social distance between a candidate and the desired characteristics

		of the Role. Between 0 and 1. The sum of homophilyWeight and roleWeight should be 1.0
lieLocationProbability	Double	The probability that an entity will lie about the location of a behavior that reveals location.

Table 7.3.59 KLEParameters

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.60 SocialNetworkBehavior

Defines behaviors that can be assigned to Roles in the dynamic social network.

Column/Field Name	Type	Description
Name	String	The name of the behavior
maxDelay	Double	An upper bound on the delay between occurrences of the behavior.
startDistribution	String	The distribution of the time delay between when an entity is assigned the behavior and when it first executes the behavior
delayDistribution	String	The distribution of the time between executions of the behavior. The value is bounded by maxDelay
behaviorInformation	String	A comma separated list of the methods of an entity that are reported when this behavior is executed. The methods should return a String or an object with a meaningful toString(). (getName and getLocation)
Type	String	Used with the KnowledgeProbability table to determine the probability of an entity in a Role learning about the behavior.

Table 7.3.60 SocialNetworkBehavior

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.61 Roles

Defines roles within the key leader and dynamic social networks. Note that behaviors only apply to the dynamic social network, not the key leader network.

Column/Field Name	Type	Description
Name	String	The name of a Role in the network.
Group	String	A group of similar Roles. Used when replacing a key leader if one with the exact Role is not available.

Rank	Double	A relative ranking of the Roles. A lower number is a higher ranking leader.
startBehavior	String	The name of the social network behavior that will be executed when an entity assumes the Role
endBehavior	String	The name of the social network behavior that will be executed when an entity leaves the Role
socialNetwork	String	The name of the social network that this Role is part of
correspondingRole	String	The Role that is the reverse of this Role. Can be blank
Active	Boolean	True if this is an active Role.
attritionTime	String	The distribution for the time until a relationship will cease to exist. (See Note 1 for format)
maxRelationships	Double	The distribution for the number of relationships for a Role. (See Note 1 for format)
maxSocialDistance	Double	The maximum social distance between the weighted average of the distance between 2 entities and the distance from the candidate to the desired attributes that will allow them to form a relationship based on the Role. Must be between 0 and 1 inclusive.
timeBetweenChanges	String	The distribution for the time between the maximum relationships being changed. (See Note 1 for format)
Type	String	The type of role. Used with the KnowledgeProbability table to determine the probability that an entity in a role will know about a behavior of another entity.
Derived	Boolean	True if this Role is derived as defined in the DerivedRelationship table.

Table 7.3.61 Roles Worksheet/Table

Note 1: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.62 RoleBehaviors

Defines which behaviors are associated with which Roles

Column/Field name	Type	Description
Role	String	The name of the Role
Behavior	String	The name of a behavior associated with the Role

Table 7.3.62 RoleBehavior Worksheet/Table

7.3.63 RoleQualification

Defines the desired or required social dimension values for a Role in the dynamic social network. For required qualifications, multiple values can be listed for a type. For desired qualifications, only one type should be listed.

Column/Field Name	Type	Description
role	String	The name of the Role
Type	String	“Required” indicates that the entry is required to form a relationship. “Desired”

		indicates that the entry is desired to form a relationship. "Disaggregate" indicates that the given social dimension must match between the entities in order to form a relationship.
Dimension	String	The name of a social dimension
valueType	String	The name of a social dimension value type (Blank if the type is "Disaggregate")

Table 7.3.63 RoleQualification Worksheet/Table

7.3.64 KeyLeaderNetwork

Holds the initial structure of the Key Leader Network

Column/Field Name	Type	Descrption
Name	String	The name of the network
Leader	String	The name of the leader.
Role	String	The Role of the leader.
Subordinate	String	A Subordinate to the leader in a Role. To add an empty Role for a leader, leave the Subordinate blank.

Table 7.3.64 KeyLeaderNetwork Worksheet/Table

7.3.65 AffinityLevels

Used to map affinity values to discrete affinity levels. Higher values indicate higher affinity.

Column/Field Name	Type	Description
Name	String	The name of the level.
lowerLimit	Double	The lower range for the level. Values equal to the lower range are in the range.
upperLimit	Double	The upper range for the level. Values equal to the upper range are out of the range.
integerValue	Integer	The value that will be recorded in PAVE for the level.

Table 7.3.65 AffinityLevels Worksheet/Table

7.3.66 AffinityNetwork

Specifies the initial affinity levels among entities. (The affinities are one way.)

Column/Field Name	Type	Description
From	String	The name entity holding the affinity
To	String	The name of the object of the affinity
Level	String	The initial affinity level.

Table 7.3.66 AffinityNetwork Worksheet/Table

7.3.67 LightEntityPrototype

Specifies how to generate LightEntities to support the dynamic social network

Column/Field Name	Type	Description
Name	String	The base name of the entities
nameFile	String	A file containing names to use for these entities. Optional, if blank, the base name is used to generate the names. More than one prototype can use the same name file.
Location	String	The location of these entities
countDistribution	String	The distribution for the number of entities to generate (See Note)

Table 7.3.67 LightEntityPrototype Worksheet/Table

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.68 LightEntitySocialDimensions

Associates social dimension data with the generated light entities.

Column/Field Name	Type	Description
entityName	String	The entity prototype name corresponding to the LightEntityPrototype table
Dimension	String	The name of a SocialDimension
valueType	String	The name of a SocialDimensionValueType for the specified SocialDimension

Table 7.3.68 LightEntitySocialDimensions Worksheet/Table

7.3.69 KnowledgeProbability Worksheet/Table

Holds the probability that an entity in a Relationship would know about a behavior by the other entity

Column/Field Name	Type	Description
RoleType	String	The type of Role
BehaviorType	String	The type of behavior
Probability	Double	The probability that an entity in a Role of a given type would know about a behavior of the given type

Table 7.3.69 KnowledgeProbability Worksheet/Table

7.3.70 SocialNetwork Worksheet/Table

Declares the social networks for the scenario. The data in the SocialNetwork in earlier versions has been moved to HomophilyNetwork.

Column/Field Name	Type	Description
Name	String	The name of the network

Table 7.3.70 SocialNetwork Worksheet/Table

7.3.71 DerivedRelationship Worksheet/Table

Holds the definitions of how new relationships are derived from existing relationships. A definition can have multiple rows in the table. The definitions are combined according the value of the "and" column. The rows for a definition are evaluated in the order entered. There is no operator precedence. The first false row with "and" true causes the definition to be false. The first true row with "and" false causes the definition to be true. If all rows are true with "and" true, then the definition is true. If all rows are false with "and" false, then the definition is false.

Column/Field Name	Type	Description
Name	String	The name of the definition.
Definition	String	The definition for the row. There are 4 possible formats discussed below.
And	Boolean	If true, then this row is "and'd" to the others, otherwise it is "or'd" See additional discussion above
Not	Boolean	If true, then the results of this row are negated.
Role	String	The name of the derived Role. The Role should be the same for all rows with the same definition name.

Table 7.3.71 DerivedRelationship Worksheet/Table

The definition entry can be in one of the following formats:
1) Role (The object of the existing relationship must have this Role. For example: Given the definition "Child", if entity1 has a Child entity2 then entity2 could be assigned the Role of dependent of entity1.)
2) Role->Attribute.Value (The object of the existing relationship must have the Role and have the given value for the given attribute. Given the definition "Child->Gender:Male, If entity2 has the Role of Child relative to entity1 and is male then entity2 could be assigned the Role of Son relative to entity1.)
3) Role.Role (The candidate entity has the second relation with a third party that has the first relationship with the subject of the existing relationship. For example: In the definition Parent.Wife, entity1 has a Parent entity2 and entity2 has a Wife enitity3 then enitity3 would be assigned the role of Mother to entity1)
4) Attribute:Value (The subject of the relationship must have the given value for the given attribute. Given the definition "Age:Mature" then if entity1 is mature, then entity1 could be assigned the role of elder relative to any other entity.)

7.3.72 Output Worksheet/Table

Specifies what reports to output during the run. Each different type of data logger requires different columns. All output will be formatted as comma-delimited, text files.

Column/Field Name	Type	Description
Type	String	The type of data logger. (See Note 1.)
Name	String	An arbitrary name that will appear in the output files to allow output lines to be matched with logger definitions.

File	String	The name of the output file that the logger will write to.
StartTime	double	The time the logger starts logging data. (See Note 2.)
Period	double	How often the logger logs data. (See Note 2.)
EntityElement	String	The type of entity to log data about. (See Note 3.)
EntityName	String	The name of the entity to log data about or “ALL” to log data about all entities. (See Note 4.)
PropertyName	String	The property to log data about. (See Note 5.)
OutputSummary	String	Enter “Yes” to output summary over all replications; otherwise, enter “No”. (See Note 6.)
LogOldValue	String	Enter “Yes” to log the previous and current value at the time of change; otherwise, enter “No”. (See Note 7.)

Table 7.3.72 Output Worksheet/Table.

Note 1: The following types of data loggers are available:

a. PositionChangeDataLogger – Logs the values of issue positions each time an agent processes the effects of an action.
b. StateDataLogger – Logs state changes of entities.
c. ConsumableChangeDataLogger – Logs the on-hand quantity of consumables whenever they are transferred.
d. ActionDataLogger – Logs actions each time they occur. Actions are collected by ActionType. (See 7.3.36, “ActionType Worksheet/Table”.)
e. BehaviorDataLogger - Logs actions each time they occur. Actions are collected by Behavior Type. (See 7.3.25, “Behavior Worksheet/Table”.)
f. HomophilyNetworkDataLogger – Periodically logs the link weights between every pair of agents representing the civilian population.
g. PositionAverageDataLogger – Periodically logs the average value of issue position changes.
h. PositionTimeAverageDataLogger – Periodically logs the time average value of issue position changes.
i. LocationDataLogger – Logs the location of an agent.
j. CountDataLogger – Logs any of the following:
    1) Logs the number of agents served when an Infrastructure finishes serving an agent.
    2) Logs the number of agents who have balked when an agent arrives at an Infrastructure, all servers are busy, and the queue is filled to capacity or is too long for the agent to tolerate.
    3) Logs the number of agents who have reneged when an agent waiting in the queue leaves the queue and departs the Infrastructure.
    4) Logs the number of agents who have arrived when an agent arrives at an Infrastructure.

k. SimpleStatsDataLogger – Log the current average wait time when an agent leaves the queue of an Infrastructure, logs the average service time when an agent finishes serving an agent, and logs the average system time when an agent departs an Infrastructure.
l. TimeVaryingStatsDataLogger – Logs the time varying average queue size and time varying average number of available servers when an Infrastructure's queue size changes and the number of available servers changes, respectively.
m. MemoryDataLogger – Logs information about Java memory usage and run time.

Note 2: Required by the HomophilyNetworkDataLogger, PositionAverageDataLogger, PositionTimeAverageDataLogger, and MemoryDataLogger.

Note 3: Required by the PositionChangeDataLogger, StateDataLogger, ConsumableChangeDataLogger, ActionDataLogger, BehaviorDataLogger, HomophilyNetworkDataLogger, PositionAverageDataLogger, PositionTimeAverageDataLogger, LocationDataLogger, CountDataLogger, SimpleStatsDataLogger and TimeVaryingStatsDataLogger.

a. For the PositionChangeDataLogger, PositionAverageDataLogger, PositionTimeAverageDataLogger, ActionDataLogger, BehaviorDataLogger and LocationDataLogger the EntityElement is "Agent".
b. For the StateDataLogger, ConsumableChangeDataLogger, CountDataLogger, SimpleStatsDataLogger and TimeVaryingStatsDataLogger, the EntityElement is "InfrastructureServer"
c. For the HomophilyNetworkDataLogger, the EntityElement is "HomophilyNetworkUmpire".

Note 4: Required by the PositionChangeDataLogger, StateDataLogger, ConsumableChangeDataLogger, ActionDataLogger, BehaviorDataLogger, HomophilyNetworkDataLogger, PositionAverageDataLogger, PositionTimeAverageDataLogger, LocationDataLogger, CountDataLogger, SimpleStatsDataLogger and TimeVaryingStatsDataLogger.

Note 5: Required by the PositionChangeDataLogger, StateDataLogger, ConsumableChangeDataLogger, ActionDataLogger, BehaviorDataLogger, HomophilyNetworkDataLogger, PositionAverageDataLogger, PositionTimeAverageDataLogger, LocationDataLogger, CountDataLogger, SimpleStatsDataLogger and TimeVaryingStatsDataLogger:

a. For the PositionChangeDataLogger, PositionAverageDataLogger, and the PositionTimeAverageDataLogger, the PropertyName takes the form "Issue-<IssuePrototype name>". For example, if the IssuePrototype name is "Land_Reform", the PropertyName is "Issue-Land_Reform" (without the quotes).
b. For the StateDataLogger the PropertyName takes the form "State-ALL" if EntityName is "ALL" or "State-<entity name>" if EntityName is the name of an entity. For example, if the EntityName is "Foo", the PropertyName is "State-Foo".
c. For the ConsumableChangeDataLogger, the PropertyName takes the form "ConsumableLevel-ALL" if EntityName is "ALL" or "ConsumableLevel-<entity name>" if EntityName is the name of an entity. For example, if the EntityName is "Foo", the PropertyName is "ConsumableLevel-Foo".

d. For the ActionDataLogger, the PropertyName takes the form “Action-<ActionType name>”. For example, if the ActionType name is “DamageInfrastructure”, the PropertyName is “Action-DamageInfrastructure”.
e. For the BehaviorDataLogger, the PropertyName takes the form “Behavior-<Behavior type name>. For example, if the behavior type is “INSURGENT_ACTION”, the PropertyName is “Behavior-INSURGENT_ACTION”. (See 7.3.25, “Behavior Worksheet/Table” for a list of behavior types.)
f. For the HomophilyNetworkDataLogger, the PropertyName is always “Homophily-ALL“.
g. For the LocationDataLogger, the PropertyName takes the form “Location-ALL” if EntityName is “ALL”, or “Location-<entity name>” if EntityName is the name of an entity. For example, if the EntityName is “Foo”, the PropertyName is “Location-Foo”.
h. For the CountDataLogger, the PropertyName can take the following forms:
    1) “NumberServed -ALL” if EntityName is “ALL”, or “NumberServed -<entity name>” if EntityName is the name of an Infrastructure.
    2) “NumberBalked-ALL” if EntityName is “ALL”, or “NumberBalked--<entity name>” if EntityName is the name of an Infrastructure.
    3) “NumberReneged-ALL” if EntityName is “ALL”, or “NumberReneged -<entity name>” if EntityName is the name of an Infrastructure.
    4) “NumberArrived-ALL” if EntityName is “ALL”, or “NumberArrived -<entity name>” if EntityName is the name of an Infrastructure.
i. For the SimpleStatsDataLogger, the PropertyName can take the following forms:
    1) “WaitTime-ALL” if EntityName is “ALL”, or “WaitTime-<entity name>” if EntityName is the name of an InfrastructureServer.
    2) “ServiceTime-ALL” if EntityName is “ALL”, or “ServiceTime-<entity name>” if EntityName is the name of an InfrastructureServer.
    3) “SystemTime-ALL” if EntityName is “ALL”, or “SystemTime-<entity name>” if EntityName is the name of an InfrastructureServer.
j. For the TimeVaryingStatsDataLogger, the PropertyName can take the following forms:
    1) “QueueSize-ALL” if EntityName is “ALL”, or “QueueSize-<entity name>” if EntityName is the name of an InfrastructureServer.
    2) “AvailableServer-ALL” if EntityName is “ALL”, or “AvailableServer-<entity name>” if EntityName is the name of an InfrastructureServer.

Note 6: Required by the PositionChangeDataLogger, PositionAverageDataLogger, PositionTimeAverageDataLogger, CountDataLogger and TimeVaryingStatsDataLogger.
Note 7: Required by the PositionChangeDataLogger.

7.3.73 PaveInterface Worksheet/Table

Provides information needed for SIM to connect to a Planning, Adjudication, and Visualization Environment (PAVE) database. Only one row of data is required. The data is described in Table 7.3.73.

Column/Field Name	Type	Description

name	String	A name for the PaveInterface. If blank, SIM will run standalone, i.e., SIM will run without connecting to PAVE.
class	String	The class name of the PaveInterface defined in the rucg.mas.twg package. (See Note 1.)
server	String	The name of the server where the PAVE database resides.
db	String	The name of the PAVE database file including the path (either relative or full). The database is expected to be either Microsoft Access or Microsoft SQL Server.
User	String	The authorized user's name to access the PAVE database. Applies only to Microsoft SQL Server; leave blank, otherwise.
passwd	String	The password if an authorized user's name is required; otherwise, leave blank
driver	String	The class name for the driver to be used for the connection. (See Note 2.)
firstRerunPauseTime	double	The SIM time at which this CgPaveInterface pauses for the first time if SIM needs to be restarted from time zero during the exercise. If this is the very first time SIM is being run during the exercise, the value of this field should be -1.
issuePosition	String	The name of the IssuePositionPrototype defined in Table 7.3.6 for which the issue stance will be summarized and written to PAVE's CG_IssueStance table.

Table 7.3.73 PaveInterface Worksheet/Table.

Note 1: Enter either "CgPaveInterface" or "rucg.mas.twg.CgPaveInterface" (without the quotes).

Note 2: Enter one of the following:

- sun.jdbc.odbc.JdbcOdbcDriver
- com.microsoft.sqlserver.jdbc.SQLServerDriver
- net.sourceforge.jtds.jdbc.Driver

7.4 Key Leader Engagement Umpire Input File.

The key leader engagement file is an XML file that controls the translation of PAVE model instructions into key leader engagement events in SIM. The sections that follow provide a description of the format of the file. A sample file is available in **Error! Reference source not found.**.

The root element of the file is the KeyLeaderEngagementUmpire. The file is inserted in the generated XML scenario file after the RoleGroup elements.

7.4.1 KeyLeaderEngagmentUmpire Element

The KeyLeaderEngagmentUmpire has one or more KLEHandler sub-elements. Its attributes are summarized in the following table.

Attribute Name	Data Type	Description
name	String	An arbitrary name for the umpire that may appear in warning and error messages.
probDetaineeKnowledge	double (0 – 1)	The probability that a detainee will give knowledge during an interview
probHumintKnowledge	double (0 – 1)	The probability that an entity will give knowledge when interviewed by intelligence personnel.
probNonHumintKnowledge	double (0 – 1)	The probability that an entity will give knowledge when interviewed by non-intelligence personnel.
probCriticalKnowledge	double (0-1)	The probability that a key leader will give any type of critical knowledge.
probPersonKnowledge	double (0 – 1) See Note	The probability that a key leader will give the name of a subordinate during a successful key leader interview.
probTransactionKnowledge	double (0 – 1) See Note	Probability that a key leader will give information during a key leader interview.

Table 7.4.1 KeyLeaderEngagementUmpire Attributes

Note: The sum of probPersonKnowledge and probTransactionKnowledge must be in the range (0-1). If neither person knowledge nor transaction knowledge is passed, then the result type provideCriticalKnowledge is used.

7.4.2 KLEHandler Element

The KLEHandler element has one or more KLEAlgorithm elements and an optional ModifierTranslator element. KLEHandlers first extract the needed data from the model instruction and pass it to the KLEAlgorithms for processing. The table below contains the attributes common to all KLEAlgorithm implementations, note 1 below the table may contain additional attributes for a given implementation.

Attribute Name	Data Type	Description
Class	String	Fully qualified class name of an implementation of KLEHandler (See Note 1 below.)
baseInstruction	String	The base name of the model instruction that the KLEHandler handles. (See Note 2 below.)
name (optional)	String	An arbitrary name that may appear in error and warning messages.

Table 7.4.2 KLEHandler Attributes

Note 1: The following implementations of KLEHandler are available. Additional user supplied implementations are allowed as long as they are on the class path.

rucg.mas.kle.hanlders.LocationHandler: Assumes that there is no specified target of the interaction and that all entities or key leaders in a location are the target of the interaction. Additionally, if an entity has any SocialDimensions that match a property set by the ModifierMatchers, then it must match the value specified. The LocationHandler has these additional attributes:

1) keyLeadersOnly (optional defaults to false): If true then only key leaders are included.

2) useModifierAsLocation (optional defaults to false): If true, then the location is derived from the model instruction using a ModifierMatcher, discussed below. If false, then the location comes from location column in the CG_Events_To_Fire table in the PAVE database.

rucg.mas.kle.handlers.KeyLeaderAsTargetHandler: Assumes that the specified key leader is the target of the interaction. Has no additional attributes.

rucg.mas.kle.handlers.NoTargetSpecifiedHandler: Assumes that any interactions are not specific to any entities or location. Has no additional attributes.

Note 2: The baseInstruction is the base part of the modeling with any modifiers removed. For example the model instructions “InterviewCivilian”, “InterviewFemale”, InterviewHeadOfHousehold”, and “HUMINTInterviewDetainee” all share the base instruction “Interview” and can be handled by the same KLEHandler. The parts of the instruction when “Interview” is removed are the modifiers and are processed by a ModifierMatcher to convert them in to properties for the algorithms to use.

7.4.3 KLEAlgorithm Element

The KLEAlgorithm Element has no common sub-elements. Elements unique to implementation of KLEAlgorithm will be discussed below. The KLEAlgorithms process the interaction.

Attribute Name	Data Type	Description
Class	String	A fully qualified class name of an implementation of KLEAlgorithm. (See Note 1 for available classes)

Table 7.4.33 KLEAlgorithm Attribute

Note 1: The following KLEAlgorithm implementations are available.

rucg.mas.kle.algorithms.InterviewAlgorithm: Implements the interview of an entity. The entity interviewed is picked randomly from the available entities. The result of the interview is the gaining of knowledge about behaviors of the entity and others. Whether the entity gives the knowledge is controlled by the probDetaineeKnowledge, probHumitKnowledge, and probNonHumingKnowledge. It uses the “HUMINT” and “Detainee” properties to determine which probability to use. The InterviewAlgorithm is usually used with a LocationHandler.

rucg.mas.kle.algorithms.KeyLeaderAttritionAlgorithm: Causes the key leader to be killed.

rucg.mas.kle.algorithms.KeyLeaderInteractionAlgorithm: Processes key leader engagements. Has KLEAction sub-elements. Uses the “BribeSize” and “Threaten” attributes supplied by the ModifierMatchers. If the engagement is successful as determined by the affinity of the leader to the initiator and the BribeSize and Threaten attributes the following occurs:

1) The Actions for any KLEActions that match the SocialDimension values specified are scheduled.

2) Based on probPersonalKnowledge, the leader may supply the name of a subordinate.

3) Base on the probTransactionKnowledge, the leader may supply knowledge about it and other entities behaviors.

Has the following additional attributes:

1) lieAffinity: The maximum affinity for the initiator that will cause the key leader to lie about having knowledge.

2) minAffinity: The minimum affinity for the initiator that will cause the engagement to be a success without threat or bribe.

rucg.mas.kle.algorithms.SigintAlgorithm: Used to pass information about behaviors collected by SIGINT.

7.4.4 ModifierTranslator Element

Has ModifierMatcher sub-elements and no attributes.

7.4.5 ModifierMatcher Element

Has Property sub-elements. Used to convert the model instruction modifiers into attributes for use by the KLEALgorithms.

Attribute Name	Data Type	Description
pattern	String	The pattern to test the model instruction against.
matchMethod	String	Used to determine how the pattern is compared to the model instruction (Note 1 contains the available methods.)

Table 7.4.5 ModifierMatcher Attributes

Note 1: The following matchMethods may be specified:

1) CONTAINS: Matches if the model instruction contains the pattern.

2) ENDSWITH: Matches if the model instruction ends with the pattern.

3) EQUALS: Matches if the model instruction exactly matches the pattern.

4) STARTSWITH: Matches if the model instruction starts the pattern.

5) REGEX: Uses the pattern as a regular expression to match the model instruction.

7.4.6 Property Element

Has no sub-elements. Determine the value of a property to set based on a ModifierMatcher matching the model instruction with the pattern.

Attribute Name	Data Type	Description
name	String	The name of the property.
value	String	The value of the property.

Table 7.4.6 Property Attributes

7.4.7 KLEAction Element

Has a KeyLeader, Initiator, and one or more Action elements.

7.4.8 KeyLeader Element

Has one or more SocialDimension sub-elements. Used to specify the SocialDimension values of the key leader that need to match for the Actions to occur.

7.4.9 Initiator Element

Has one or more SocialDimension sub-elements. Used to specify the SocialDimension values of the initiator that need to match for the Actions to occur.

7.4.10 Action Element

Has no sub-elements.

Attribute	Data Type	Description
Name	String	The name of an Action to take.

Sample Script to Launch SIM

The script below is from the Windows XP/Vista batch file sim2.bat that is included with the downloaded source code. This file should be located in the SIM home directory.

```
 1 @REM A script to run SIM (Social Impact Module) v2 multi-agent system
standalone with no setup.
 2 @REM $Id: sim2.bat 1235 2012-09-11 00:06:26Z hmyamauc $
 3 @REM John Ruck (Rolands and Associates Corporation 11/11/05)
 4 @REM
 5 @REM Assumes Java 1.6 or later is installed and on the PATH.
 6 @Rem SIM home directory
 7 @set SIMHOME=.
 8
 9 @Rem Class and jar file locations
10 @set MYCLASSES=%SIMHOME%\build\classes
11 @set MYLIB=%SIMHOME%\lib
12
13 @Rem Input directory
14 @set MYDATADIR=%SIMHOME%\data\mas\
15 @set MYDATAFILE=%MYDATADIR%%1
16
17 @Rem Needed jar files
18 @set SIMJAR=%MYLIB%\sim2.jar
19 @set OPENOFFICEJAR=%MYLIB%\hsqldb.jar
20 @set SIMKITJAR=%MYLIB%\simkit.jar
21 @set NPSJAR=%MYLIB%\NpsTracCommon.jar
22 @set JDOMJAR=%MYLIB%\jdom.jar
23 @set JDOMCJAR=%MYLIB%\jdom-contrib.jar
24 @set TRACBAYESJAR=%MYLIB%\tracBayes.jar
25 @set WEKAJAR=%MYLIB%\weka.jar
26 @set COORDJAR=%MYLIB%\coordconv.jar
27 @set GTGEOMETRYJAR=%MYLIB%\gt-geometry-2.7.1.jar
28 @set JTDSJAR=%MYLIB%\jtds-1.2.5.jar
29
30 @Rem Set CLASSPATH
31 @set CLASSPATH=.;%
SIMJAR%;%NPSJAR%;%SIMKITJAR%;%OPENOFFICEJAR%;%JDOMJAR%;%JDOMCJAR%;%TRACBAYESJAR
%;%WEKAJAR%;%COORDJAR%;%GTGEOMETRYJAR%;%JTDSJAR%
32
33 java -Xmx1024m -cp %CLASSPATH% -
Djava.util.logging.config.file=.\logging.properties -XX:+UseParallelGC
rucg.mas.main.RucgMain %MYDATAFILE%
```

The batch file requires one argument: the name of the input file. Since the batch file is set up to read input files from the data\mas directory (line 14), there is no need to include the path within the argument. There is an example input file in XML format in this directory: example.xml. To run SIM from the command line using sim2.bat with the input file example.xml in the data\mas directory, go to the SIM home directory and type

```
sim2 example.xml
```

Line 33 illustrates four java application launcher command line options typically used to run SIM:

- `-Xmx1024m` sets the memory allocation to 1 gigabyte,
- `-cp %CLASSPATH%` specifies the directories and jar files to search for class files,
- `-Djava.util.logging.config.file=.\logging.properties` indicates that the logging facilities are to be configured based on the file logging.properties located in the SIM home directory,
- `-XX:+UseParallelGC` – enables garbage collection on multiple threads. Use this option when running SIM on a system with multiple processors.

As SIM runs, it writes output specified in 7.3.72, “Output Worksheet/Table”, to comma-delimited files it creates in the SIM home directory. In addition, the logging facilities will collect information, warning and error messages (as specified by logging.properties) and save them to a text file called java0.log.0. Again, this file will be created in the home directory. Running example.xml should only generate information and warning messages.

The sim2.bat file can easily be modified to suit a given scenario. At a minimum, the value on the right-hand side of line 14 should be modified to point to the location of the scenario data correctly. The memory allocation on line 31 may have to be increased depending upon the number of agents in the scenario. For example, SIM runs on a 32-bit system with about 200 agents in the scenario when 1 GB of memory is allocated. Increasing the number of agents to 1000 will probably require that SIM be run on a 64-bit system and memory allocation increased from 1 gigabyte to 2 gigabytes.

Sample Script to Convert Scenario Workbook to XML

The script below is from the Windows XP/Vista batch file convert.bat that is included with the downloaded source code. This file should be located in the SIM home directory.

```
 1 @REM A script to convert SIM xlsx scenario to xml.
 2 @REM $Id: convert.bat 1241 2012-09-13 16:56:03Z hmyamauc $
 3 @REM John Ruck (Rolands and Associates Corporation 11/11/05)
 4 @REM
 5 @REM Assumes Java 1.6 or later is installed and on the PATH.
 6 @Rem SIM home directory
 7 @set SIMHOME=.
 8
 9 @Rem Class and jar file locations
10 @set MYCLASSES=%SIMHOME%\build\classes
11 @set MYLIB=%SIMHOME%\lib
12
13 @Rem Input directory
14 @set MYDATADIR=%SIMHOME%\data\mas\
15 @set MYDATAFILE=%MYDATADIR%%1
16
17 @Rem Needed jar files
18 @set SIMJAR=%MYLIB%\sim2.jar
19 @set OPENOFFICEJAR=%MYLIB%\hsqldb.jar
20 @set SIMKITJAR=%MYLIB%\simkit.jar
21 @set NPSJAR=%MYLIB%\NpsTracCommon.jar
22 @set JDOMJAR=%MYLIB%\jdom.jar
23 @set JDOMCJAR=%MYLIB%\jdom-contrib.jar
24 @set TRACBAYESJAR=%MYLIB%\tracBayes.jar
25 @set WEKAJAR=%MYLIB%\weka.jar
26 @set COORDJAR=%MYLIB%\coordconv.jar
27
28 @Rem Set CLASSPATH
29 @set
CLASSPATH=.;%SIMJAR%;%NPSJAR%;%SIMKITJAR%;%OPENOFFICEJAR%;%JDOMJAR%;%JDOMCJAR%;
%TRACBAYESJAR%;%WEKAJAR%;%COORDJAR%
30
31 @Rem cluster
32 @Rem java -Xmx1024m -cp %CLASSPATH% -33
Djava.util.logging.config.file=.\logging.properties -XX:+UseParallelGC
rucg.mas.main.RucgMain %MYDATAFILE% convert noInputPath
34 @Rem standalone workstation
35 java -Xmx1024m -cp %CLASSPATH% -
Djava.util.logging.config.file=.\logging.properties -XX:+UseParallelGC
rucg.mas.main.RucgMain %MYDATAFILE% convert
```

The script is generally set up similarly to sim2.bat in Appendix A. Note that the line that launches SIM (line 35) has an extra argument, “convert”, that tells SIM to convert MYDATAFILE to XML and to stop immediately after the conversion is completed. Any information, warning or error messages will be collected and saved to a text file called java0.log.0 in the SIM home directory.

The batch file requires one argument: the name of the input file such as an Excel .xlsx file. Since the batch file is set up to read input files from the data\mas directory (line 14), all Bayesian network files and case files referenced by the input file must reside in this directory along with the input file, itself. To run from the command line using convert.bat, go to the SIM home directory and type

```
convert inputfile
```

where inputfile is the name of the input spreadsheet (.xls, .xlsb, .xlsm, .xlsx) or database (.accdb, .mdb, .odb) to be converted. The name should not include the path.

The convert.bat file can be easily modified to point to a different location by changing the directory on the right-hand side of line 14.

Specifying Weka or LightBayes

SIM interfaces to Bayesian networks through the TracBayes API. The API was developed to provide a common interface that will allow an application to inter-operate with three Bayesian network packages: Netica-J API Library (version 4.03), Weka 3, and LightBayes. TracBayes includes a properties file, bayesNetFactory.properties, which instructs the API how to interface with any of the three network packages based on the file extension of the Bayes net file that the application reads. The default settings of the properties file are shown below.

```
 1 #
 2 # Holds a mapping from file extension to the name of the BayesNetCreator
 3 # that can read that extension type. The file extension is not case
 4 # sensative.
 5 #
 6 # Version: $Id$
 7 # Author: John Ruck (Rolands and Associates Corporation) 2/2/09
 8 #
 9     xml=edu.nps.trac.tracBayes.bbn.weka.WekaCreator
10     neta=edu.nps.trac.tracBayes.bbn.netica.NeticaCreator
11     dne=edu.nps.trac.tracBayes.bbn.lightBayes.LightBayesCreator
```

The default settings instruct TracBayes to use Weka if the Bayes net file has an .xml extension, Netica if the file extension is .neta, and LightBayes if the file extension is .dne. These settings can be changed by simply changing the Creator associated with the file extension. For example, to change the association of .dne files from LightBayes to Weka, change line 11 to

```
dne=edu.nps.trac.tracBayes.bbn.weka.WekaCreator
```

The bayesNetFactory.properties file is included with the SIM source code and is located in the src directory.

Warning: It is not advised to use Netica with SIM because Netica will (unpredictably) throw an Allocation error. The trace stack indicates that the error is always thrown from within Netica.dll. It is unpredictable because a given SIM scenario could be run on a given workstation without an Allocation error thrown one day, but that same scenario could be run on the same workstation the next day and an Allocation error may be thrown during the first replication. This problem has not been experienced with LightBayes and Weka, therefore it is recommended that SIM be used with only these two packages.

Sample Key Leader Engagement Umpire Input File.


```
<!-- $Id: engagementUmpire.xml 231 2012-07-11 17:11:55Z jlruck $ -->
<KeyLeaderEngagementUmpire name="KLEUmpire" probDetaineeKnowledge="1.0"
probHumintKnowledge="1.0" probNonHumintKnowledge="1.0"
probPersonKnowledge="1.0" probTransactionKnowledge="1.0">
<!-- For handling LeaderAttrition -->
  <KLEHandler class="rucg.mas.kle.handlers.KeyLeaderAsTargetHandler"
baseInstruction="LeaderAttrition" name="LeaderAttritionHandler">
    <KLEAlgorithm class="rucg.mas.kle.algorithms.KeyLeaderAttritionAlgorithm"
name="LeaderAttritionAlgorithm"/>
  </KLEHandler>
<!-- InterviewCivilian InterviewFemale InterviewHeadOfHousehold
HUMINTInterviewDetainee -->
  <KLEHandler class="rucg.mas.kle.handlers.LocationHandler"
baseInstruction="Interview" >
    <ModifierTranslator>
      <ModifierMatcher matchMethod="ENDSWITH" pattern="HeadOfHousehold">
        <Property name="Gender" value="Male"/>
        <Property name="Age" value="Middle"/>
      </ModifierMatcher>
      <ModifierMatcher matchMethod="ENDSWITH" pattern="Female">
        <Property name="Gender" value="Female"/>
      </ModifierMatcher>
      <ModifierMatcher matchMethod="STARTSWITH" pattern="HUMINT">
        <Property name="HUMINT" value="true"/>
      </ModifierMatcher>
      <ModifierMatcher matchMethod="ENDSWITH" pattern="Detainee">
        <Property name="Detainee" value="true"/>
      </ModifierMatcher>
    </ModifierTranslator>
    <KLEAlgorithm class="rucg.mas.kle.algorithms.InterviewAlgorithm"/>
  </KLEHandler>
<!-- KeyLeaderInteractionBribeSmall KeyLeaderInteractionBribeMedium
KeyLeaderInteractionBribeLarge KeyLeaderInteractionThreaten -->
  <KLEHandler class="rucg.mas.kle.handlers.KeyLeaderAsTargetHandler"
baseInstruction="KeyLeaderInteractionEvent">
    <ModifierTranslator>
      <ModifierMatcher matchMethod="ENDSWITH" pattern="BribeSmall">
        <Property name="BribeSize" value="Low"/>
      </ModifierMatcher>
      <ModifierMatcher matchMethod="ENDSWITH" pattern="BribeMedium">
        <Property name="BribeSize" value="Medium"/>
      </ModifierMatcher>
      <ModifierMatcher matchMethod="ENDSWITH" pattern="BribeLarge">
```

```
            <Property name="BribeSize" value="High"/>
          </ModifierMatcher>
          <ModifierMatcher matchMethod="ENDSWITH" pattern="Threaten">
            <Property name="Threaten" value="true"/>
          </ModifierMatcher>
        </ModifierTranslator>
        <KLEAlgorithm class="rucg.mas.kle.algorithms.KeyLeaderInteractionAlgorithm"
minAffinity="Level5" lieAffinity="Level1">
          <KLEAction>
            <KeyLeader>
              <SocialDimension dimension="ISAFStance" value="PROISAF"/>
            </KeyLeader>
            <Initiator>
              <SocialDimension dimension="ISAFStance" value="PROISAF"/>
            </Initiator>
            <Action name="ProCFLeaderCampaignsForProCGStances"/>
          </KLEAction>
          <KLEAction>
            <KeyLeader>
              <SocialDimension dimension="ISAFStance" value="ANTIISAF"/>
            </KeyLeader>
            <Initiator>
              <SocialDimension dimension="ISAFStance" value="PROISAF"/>
            </Initiator>
            <Action name="AntiCFLeaderCampaignsForProCGStances"/>
          </KLEAction>
        </KLEAlgorithm>
    </KLEHandler>
<!-- SIGNINGRequest -->
    <KLEHandler class="rucg.mas.kle.handlers.NoTargetSpecifiedHandler"
baseInstruction="SIGINTRequest">
        <KLEAlgorithm class="rucg.mas.kle.algorithms.SigintAlgorithm"/>
    </KLEHandler>
<!-- ShuraRequestAco ShuraRequestBco ShuraRequestCco ShuraRequestDco -->
    <KLEHandler class="rucg.mas.kle.handlers.LocationHandler"
useModifierAsLocation="true" baseInstruction="ShuraRequest"
keyLeadersOnly="true">
        <ModifierTranslator>
          <ModifierMatcher matchMethod="ENDSWITH" pattern="ACo">
            <Property name="Location" value="Location1"/>
          </ModifierMatcher>
          <ModifierMatcher matchMethod="ENDSWITH" pattern="BCo">
            <Property name="Location" value="Location2"/>
          </ModifierMatcher>
          <ModifierMatcher matchMethod="ENDSWITH" pattern="CCo">
            <Property name="Location" value="Location3"/>
```

```
            </ModifierMatcher>
            <ModifierMatcher matchMethod="ENDSWITH" pattern="DCo">
                <Property name="Location" value="Location4"/>
            </ModifierMatcher>
        </ModifierTranslator>
        <KLEAlgorithm class="rucg.mas.kle.algorithms.KeyLeaderInteractionAlgorithm"
minAffinity="Level5" lieAffinity="Level1"/>
    </KLEHandler>
</KeyLeaderEngagementUmpire>
```

H.2. SIM 2.0a USER GUIDE

This section highlights the portions of the SIM 2.0 User Guide that changed for SIM 2.0a, the alternate OAB method (counter). There are no new worksheets in the scenario file, and SIM 2.0a eliminates the need for the AttitudePositionPrototype worksheet that defined the BBN for OABs in SIM 2.0.

The following list outlines the changes in IM 2.0a scenario file worksheets:

- AgentAttitudes - 1 column added: initialAttitudeClass
- AgentNets - 4 columns removed: attitudeNetFile, initialCaseFile2, weight2 and attitudeSelectCycleClass
- AgentBehaviors - 2 columns removed: initialCaseFile and weight
- ScriptedAttitudeEffects - contains 4 columns: index, initiator, receiver and effect
- AttitudeActionEffects - contains 7 columns: index, initiator, receiver, consumableType, providerAssociation, outcome and effect

The following pages from the SIM 2.0a User Guide show the changes. The complete document resides in the “doc” folder of the source code for SIM 2.0a.

Social Impact Module (SIM) Version 2a User Manual

TRADOC Analysis Center

29 September 2012

agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.14 whose “isExternal” field is “FALSE”.
beliefPrototype	String	The name of the BeliefPrototype defined in Table 7.3.3.

Table 7.3.16 AgentBeliefs Worksheet/Table.

7.3.17 AgentIssues Worksheet/Table

Defines issues important to AgentPrototype. One row is filled out for each issue important to an AgentPrototype as shown in Table 7.3.17.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.14 whose “isExternal” field is “FALSE”.
issuePrototype	String	The name of the IssuePrototype defined in Table 7.3.5.

Table 7.3.17 AgentIssues Worksheet/Table.

7.3.18 AgentAttitudes Worksheet/Table

Defines attitudes held by AgentPrototype. One row is filled out for each attitude held by an AgentPrototype as shown in Table 7.3.18.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.14 whose “isExternal” field is “FALSE”.
attitudePrototype	String	The name of the AttitudePrototype defined in Table 7.3.7.
initialAttitudeClass	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used for initializing the attitude called *attitudePrototype* of all agents of type *agentPrototype* at the start of each replication. (See Note.)

Table 7.3.18 AgentAttitudes Worksheet/Table

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.19 Agent Worksheet/Table

Defines each agent to be instantiated in the scenario. One row of data is entered for each agent as shown in Table 7.3.19.

		beliefs, values and interests. (See Notes 2 and 3.)
prototype	String	The name of the IssuePrototype defined in Table 7.3.5, or the name of the AttitudePrototype defined in Table 7.3.7, or "MULTIPLE". (See Note 4.)

Table 7.3.22 BayesNetFiles Worksheet/Table.

Note 1: The package name "rucg.mas.bayesnet" may be optionally prefixed to the class entry. Therefore, "IssueNet", "rucg.mas.bayesnetIssueNet", AttitudeNet and "rucg.mas.bayesnet.AttitudeNet" are legitimate entries (without the quotes).

Note 2: If SIM is running Weka or LightBayes, the Bayesian network file must be in text format following the DNET file specification (*.dne).

Note 3: All Bayesian network files must be in the same directory as the Excel, Access, or Open Office file containing the scenario data. Do not enter the full path name of the network file.

Note 4: If the file specified by *bayesNetFile* addresses only a single issue, enter the IssuePrototype name. The name must match the name of the node representing the issue. If the file addresses more than one issue, enter "MULTIPLE". Likewise, if the file specified by *bayesNetFile* addresses only a single attitude, enter the AttitudePrototype name. The name must match the name of the node representing the attitude. If the file addresses more than one attitude, enter "MULTIPLE".

7.3.23 AgentNets Worksheet/Table

This is used to declare the issues and attitudes that are relevant to each agent. It is also used to set the agent's initial positions on the issues and attitudes. Each row is filled out according to Table 7.3.23.

Column/Field Name	Type	Description
Agent	String	The name of the agent defined in Table 7.3.19.
issueNetFile	String	The name of the Bayesian network file declared in Table 7.3.22 for assessing positions on issues.
initialCaseFile1	String	The name of a case file defined in Table 7.3.21 or "NONE". (See Note 1). The file holds this agent's initial findings for the issue(s) represented by *issueNetFile*. This file is assumed to be in comma-delimited format (*.csv). (See Note 2.)
weight1	double	The weight applied to *initialCaseFile1*. It should have a positive value. (See Note 3.)

Table 7.3.23 AgentNets Worksheet/Table.

Note 1: Enter the name of a case file if learning is required to determine the initial probabilities of the conditional probability tables in the *NetFile*. Enter “NONE” if the conditional probability tables in the *NetFile* were created manually by a subject matter expert, or have already been learned well.

Note 2: If a case file name is entered, the case file must be in the same directory as the Excel, Access, or Open Office file containing the scenario data. Do not enter the full path name of the case file.

Note 3: The weight, also called degree, represents the multiplicity of the case(s) in *initialCaseFile*. A positive value of *w* is used to tell the Bayesian network to learn *w* cases at once. A negative value of –*w* is used to tell the network to “unlearn” *w* cases at once. It is assumed, however, that if an initial case file is specified, it will be used to train the network to obtain the initial beliefs, values, interests, and positions on an issue. Therefore, a negative weight should never be entered in this worksheet. There is no effect on the network if $w = 0$. The weight normally is 1.

7.3.24 Behavior Worksheet/Table

Declares the planned behaviors. Each row defines a behavior. The data is described in Table 7.3.24.

The behaviors for each agent are stored in a Map structure keyed to the name of the behavior. Therefore, each behavior must be identified by a unique name, even if the behaviors are the same “behaviorType”.

The network file only contains the structure of the Bayesian network. The structure, itself, is based on Icek Aizen’s Theory of Planned Behavior[5] (See 8, “Bayesian Networks for Simulating Planned Behaviors”). The initial state of the network is set by the AgentBehaviors worksheet described in 7.3.30, “AgentBehaviors Worksheet/Table”.

Column/Field Name	Type	Description
name	String	The name of the behavior.
behaviorType	String	Enter “COALITION_ACTION”, “GOVERNMENT_ACTION”, “INFORMANT_ACTION”, “INFRASTRUCTURE”, “INSURGANT_ACTION”, “MASS_MEDIA”, “KLE_ACTION” or “COMMUNICATE”. (See Note 1.)
consumableType	String	If *behaviorType* is “INFRASTRUCTURE”, the name of the ConsumableType from Table 7.3.13 that will be restocked. Ignored if *behaviorType* is not “INFRASTRUCTURE”.

Table 7.3.24 Behavior Worksheet/Table.

[5] Theory of Planned Behavior home page: http://people.umass.edu/aizen/tpb.html

level	String	The name of the level declared in Table 7.3.28. The entries in this field and *method* must be consistent with the entries in the "name" and "level" fields in Table 7.3.28.
intentNodeState	String	The name of the state from the node representing the Intention to perform the behavior. The entries in this field and *behavior* must be consistent with the entries in the "behaviorName" and "intentNodeState" fields in Table 7.3.25.

Table 7.3.29 BehaviorMethodAction Worksheet/Table

7.3.30 AgentBehaviors Worksheet/Table

Declares the planned behaviors that each agent will simulate. It sets the initial state of the agent's behavior, the method the agent uses to select the action it will take, and determines how frequently the behaviors are carried out. Each row is filled out according to Table 7.3.30.

Column/Field Name	Type	Description
agent	String	The name of the agent defined in Table 7.3.19.
behaviorName	String	The name of the behavior declared in Table 7.3.24.
utilityBehavior	String	If applicable, the name of the UtilityBehavior defined in Table 7.3.26.
consumableType	String	If the "behaviorType" field in Table 7.3.24 is "INFRASTRUCTURE" for *behaviorName*, the name of the ConsumableType from Table 7.3.13 that will be restocked.
initialExecuteTime	String	Enter "NONE" or the distribution to generate the (simulation) time that this behavior will be executed for the first time. (See Notes 1 and 2.)
executeInterval	String	Enter "NONE" or the distribution to generate a waiting time before this behavior is repeated. (See Notes 1 and 2.)
stopBehaviorTime	String	Enter the distribution to generate the (simulation) time to stop this behavior. Enter "NONE" if this behavior should never be stopped. (See Notes 2 and 3.)
intentSelection	String	Enter "DRAW", "HIGHEST", or "THRESHOLD". (See Note 4.)
threshold	double	If intentSelection is "THRESHOLD", the threshold value in the range [0.0, 1.0].

Table 7.3.30 AgentBehaviors Worksheet/Table.

Note 1: Enter "NONE" if the "behaviorType" field in Table 7.3.24 is either "INFRASTRUCTURE" or "COMMUNICATE" for *behaviorName*; otherwise, enter the distribution according to Note 2.

Note 2: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 3: If a behavior is stopped it cannot be re-started until the start of the next replication.

Note 4: The agent can use one of three methods to choose an intention node state, i.e., choose the action it will perform:

a. DRAW – Perform a probability draw.
b. HIGHEST – Choose the state with the highest probability.
c. THRESHOLD – Declare a threshold value in the range [0.0, 1.0] and all states whose probability exceeds this value will be executed.

7.3.31 Location Worksheet/Table

Defines geographic areas within the AO. Locations are referenced by agents/groups, infrastructure, and actions. These areas are assumed to be polygons. One row of data is entered for each location as shown in Table 7.3.31.

Column/Field Name	Type	Description
name	String	The name of the location.
Level	String	Determines what level the Location is in the hierarchy of locations.
class	String	The class name of a Location defined in the rucg.mas.location package. (See Note 1.)
coordinate	String	The center coordinate of this location. Required only if *class* is "HexLocation" or "rucg.mas.location.HexLocation". The format depends upon the coordinate system. (See Note 2.) Ignored if class is neither "HexLocation" nor "rucg.mas.location.HexLocation".
numberVertices	int	The number of vertices this location owns. Enter a positive value only if there is a need to find a location given a coordinate; otherwise enter zero.
vertexCoordinate1, vertexCoordinate2, etc.	String	If *numberVertices* is positive, enter the first vertex coordinate under vertexCoordinate1, the second vertex coordinate under vertexCoordinate2, and so on. The format depends upon the coordinate system. (See Note 2.) Ignored if *numberVertices* is zero.

Table 7.3.31 Location Worksheet/Table.

Note 1: The following Location classes are currently implemented:

7.3.38 ScriptedAttitudeEffects Worksheet/Table

Defines the effects of a ScriptedAction on an agent's or group's set of beliefs, values, interests and attitudes. Each attitude (with its supporting beliefs, values and interests) is maintained in a Bayesian network and each effect is represented by a case file. One row must be filled out for each effect as shown in Table 7.3.38.

Column/Field Name	Type	Description
index	Int	The index number of an action defined in Table 7.3.36. The number links this effect with the action.
initiator	String	The name of an AgentPrototype defined in Table 7.3.14. This is the AgentPrototype that conducted the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.14. This is the AgentPrototype that receives this effect.
effect	int	The effect of the action referenced by *index*. It should be -1, 0 or 1 where -1 indicates a negative effect, 0 indicates no effect, and 1 indicates a positive effect.

Table 7.3.38 ScriptedAttitudeEffects Worksheet/Table.

Note: The distribution is used to generate a probability. If the value generated by "priorDistribution" is less than zero (0.0), SIM sets the probability to 0.0; likewise, if the value generated by "priorDistribution" is greater than one (1.0), SIM sets the probability to 1.0. The distribution is specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses, e.g., Triangle(0.4, 0.7, 0.6).

7.3.39 BehaviorAction Worksheet/Table

Defines external operations initiated by an agent or group based on a planned behavior. One row must be filled out for each action as shown in Table 7.3.39.

The result of a BehaviorAction (either success or failure) may have an effect on an entity's set of beliefs, values and interests that, in turn, affects that entity's positions on issues and attitudes. The effects on beliefs, values and interests that affect positions on issues are entered in the IssueActonEffects worksheet described in 7.3.40, "IssueActionEffects Worksheet/Table". The effects on beliefs, values and interests that affect attitudes are entered in the AttitudeActionEffects worksheet described in 7.3.41, "AttitudeActionEffects Worksheet/Table".

Column/Field Name	Type	Description
index	int	A number used to identify this action. This number is used with the "index" column in the IssueActionEffects and AttitudeActionEffects worksheets to link the action with its effect(s).

		"behaviorType" field in Table 7.3.24 is "INFRASTRUCTURE" for "behaviorName", enter the name of an AgentPrototype defined in Table 7.3.14 whose "isExternal" field is "TRUE". Ignored if "behaviorType" is not "INFRASTRUCTURE".
outcome	String	Enter either "SUCCESS" or "FAIL". This is the outcome of the action referenced by *index*.
receiverAttitude	String	Enter "NEGATIVE", "NEUTRAL" or "POSITIVE". This is *receiver*'s attitude towards *providerAssociation* at the time this effect is received, if applicable; otherwise this is *receiver*'s attitude towards *initiator*.
priorDistribution	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. The distribution should only generate values in the range [0.0, 1.0]. (See Note.)

Table 7.3.40 IssueActionEffects Worksheet/Table.

Note: The distribution is used to generate a probability. If the value generated by "priorDistribution" is less than zero (0.0), SIM sets the probability to 0.0; likewise, if the value generated by "priorDistribution" is greater than one (1.0), SIM sets the probability to 1.0. The distribution is specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses, e.g., Triangle(0.4, 0.7, 0.6) .

7.3.41 AttitudeActionEffects Worksheet/Table

Defines the effects of a BehaviorAction on an agent's or group's set of beliefs, values, and interests. Each effect is represented by a draw from a distribution. One row must be filled out for each effect as shown in Table 7.3.41.

Column/Field Name	Type	Description
index	int	The index number of an action defined in Table 7.3.39. The number links this effect with the action.
initiator	String	The name of an AgentPrototype defined in Table 7.3.14. This is the AgentPrototype that conducted the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.14. This is the AgentPrototype that receives this effect.
consumableType	String	Match *index* from this table with the index number in Table 7.3.39 and obtain the associated "behaviorName" from Table 7.3.39. If the "behaviorType" field in Table 7.3.24 is "INFRASTRUCTURE" for "behaviorName",

		enter the name of the ConsumableType that “behaviorName” is used to restock. Ignored if “behaviorType” is not “INFRASTRUCTURE”.
providerAssociation	String	Match *index* from this table with the index number in Table 7.3.39 and obtain the associated “behaviorName” from Table 7.3.39. If the “behaviorType” field in Table 7.3.24 is “INFRASTRUCTURE” for “behaviorName”, enter the name of an AgentPrototype defined in Table 7.3.14 whose “isExternal” field is “TRUE”. Ignored if “behaviorType” is not “INFRASTRUCTURE”.
outcome	String	Enter either “SUCCESS” or “FAIL”. This is the outcome of the action referenced by *index*.
effect	int	The effect of the action referenced by *index*. It should be -1, 0 or 1 where -1 indicates a negative effect, 0 indicates no effect, and 1 indicates a positive effect.

Table 7.3.41 AttitudeActionEffects Worksheet/Table.

7.3.42 SimpleActionUmpire Worksheet/Table

Provides rules for how the SimpleActionUmpire operates. Only one row of data is required. The data is described in Table 7.3.42.

Column/Field Name	Type	Description
name	String	The name of the umpire.
recipientsNoTarget	int	The number of agents to choose at random who will receive the effects of an action if that action does not specify a target.
recepientsInfra	int	The number of agents to choose at random who will receive the effects of an action if that action specifies an infrastructure target.
doNotPassInterval	double	A period of time during which an agent will only pass an action once to other agents in its social network. (See Note 1.)
sociabilityMethod	String	Enter “K_NEAREST_NEIGHBOR” or “K_TRIM_THRESHOLD”. (See Note 2.)
sociabilityClass	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. The distribution must be consistent with the *sociabilityMethod*. (See Note 2.)
commOrder	String	Enter “DESCENDING_ORDER” or “RANDOM_ORDER”. (See Note 3.)
spatialMethod	String	Enter “COLLOCATION” or PROXIMITY. (See

APPENDIX I. SIM 3.0 TECHNICAL DESIGN CHANGE

As mentioned in the chapter on SIM 3.0, one of the goals of SIM transition was to make results in the model more explainable and traceable. The use of Bayesian Belief Networks often make it difficult to explain results from the model. Another option explored in SIM 3.0 was the data-driven approach to scenario development investigated during the Africa KDAE project sponsored by CAA. This process produces a set of equations. Those equations, combined with SME input (See Appendix J), produce a look-up table for determining agent issue stances and OAB.

This only changed one component of code within SIM - the LongTermMemory Component (Figure I.1). Appendix C provided an overview of Long Term Memory in SIM. Version 3.0 only changes the UpdateLongTermMemory node. Instead of sending values to the BBN, this node simply refers to the look-up tables and sets the new values for issue stances and OAB accordingly.

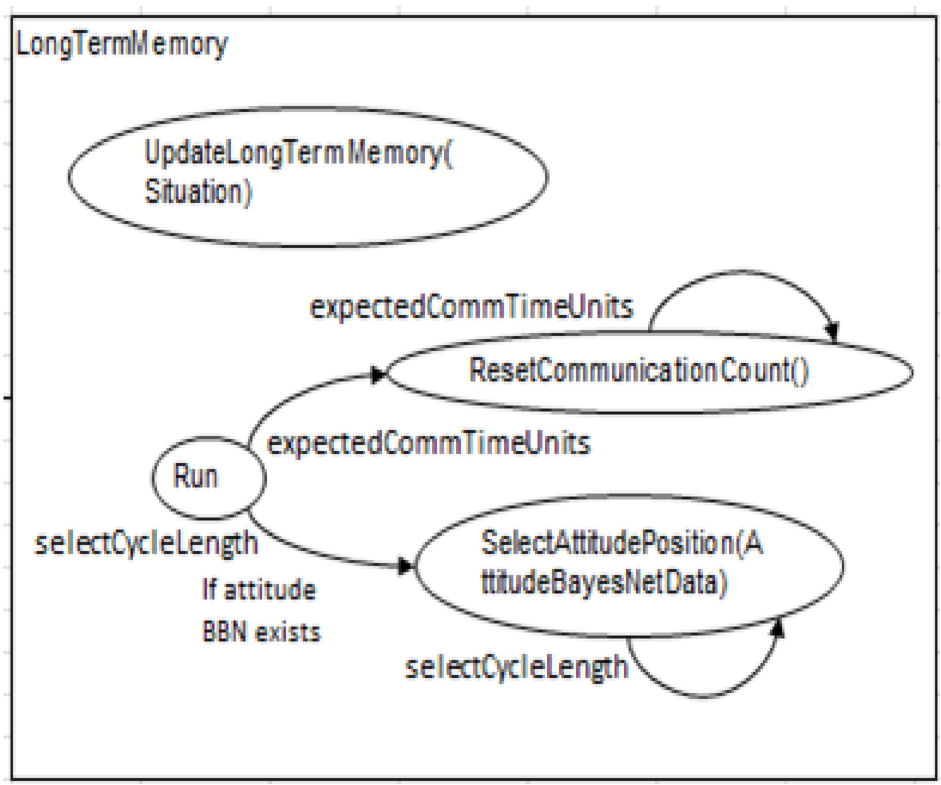

Figure I.1: Long Term Memory Component in SIM 3.0

The heart of SIM 3.0 lies in the data development process covered in Appendix J. Agents still communicate and seek infrastructure the same way they did in SIM 1.0 and SIM 2.0. If an agent communicates to another agent about an event, the effects on the agent receiving the communications would come from the look-up tables as well. Expected outcomes from SIM 3.0 can be known before the model ever runs. SIM 3.0 is a deterministic model for the scenario event effects.

In fact, IW TWG team could use the look-up tables to run a TWG as a table-top game. Another option would be to incorporate the tables into PAVE for updating issue stances and OAB. The primary need for SIM 3.0 is for communications and infrastructure outcomes. These outcomes are still stochastic. If the TWG does not require communications and infrastructure then SIM could be eliminated from the list of models required to execute the wargame if the team develops data using the techniques explored in SIM 3.0.

APPENDIX J. SIM 3.0 SCENARIO DEVELOPMENT PROCESS

This appendix comes from TRAC Technical Report, TRAC-TR-12-037, Africa Knowledge, Data Source, and Analytic Effort (KDAE) Exploration, dated 20 August 2012. The KDAE report details work sponsored by the Center for Army Analysis. Authors include MAJ Thomas Deveans, Ms. Sara Lechtenberg-Kasten, Dr. Samuel Buttrey, Dr. Ronald Fricker, Dr. Jeffrey Appleget, and LCDR Walter Kulzy. The sections of the KDAE technical report in this appendix directly support SIM Transition, specifically data development techniques used in SIM 3.0. Those sections include Chapter 4 and Appendix C-E.

SECTION 4. SCENARIO METHODOLOGY & DEVELOPMENT

4.1 INTRODUCTION

The objective of this part of this effort is to develop a methodology and build a proof of principle scenario in a specific region or country in the AFRICOM AOR for use in future IW TWG's using Factor Analysis and Generalized Linear Models. This section will describe the survey data that was used, the recoding/imputation and factor analysis, and the subsequent linear and multiple logistic regression models that will allow us to predict future population Issue Stance Scores as well as Observed, Attitudes, and Behaviors. Additionally, a small "proof of principle" scenario will demonstrate how these models can be used to predict future population responses.

4.2 THE SURVEY DATA

This part of the project is based on survey data collected in six countries in the Western Trans-Sahel region of Africa. The surveys have been conducted over the past four years, though not every country was surveyed in each of the available years. The analysis for this portion of the effort focuses on the survey conducted in the country of Nigeria, during the year 2010. This particular country was chosen as it represents a possible and likely location for the upcoming Irregular Warfare Tactical Wargame scenario lead by the TRADOC Analysis Center – White Sands Missile Range.

These surveys were initially sponsored by AFRICOM, and conducted by a private contractor operating in the region with no discernible affiliation to the U. S. military or the U.S. government. AFRICOM's objective in conducting this project was to better ascertain how their actions affect the daily lives of the indigenous populations, while also looking to identify areas of the data that can be used when determining future courses of actions or allocations of resources (Kulzy, 2012).

The survey instrument for 2010 consists of 255 questions and 3,770 respondents for the country of Nigeria. However, of these questions, some are specific to only one or two countries. There are also questions to which a Likert scale value cannot be associated, so they are coded as nominal values. There are also a number of questions that were conditional on responses to other questions. These conditional questions are, for example, specific to only one type of religion or are only answered if a previous question was affirmatively answered. These types of questions were omitted from the analyses, as they were deemed to bias the responses as they applied to only a subset of the population surveyed (Kulzy, 2012).

4.3 RECODING AND DATA IMPUTATION

Table 2 specifies the particular survey questions that were used in the analysis. All questions in the survey instrument that were asked of all respondents were included in the analysis. Conditional questions, based on skip questions, as discussed above, were not used in the analysis.

Source of Information	Q5
Quality of Life	Q6 – Q10
View of foreign countries	Q12, Q14, Q16, Q17, Q21 – Q23
Views of Nigeria	Q25
Trust and Religion	Q26 – Q34, Q36, Q37
Governance, Politics, and Security	Q40, Q45, Q48 – Q50, Q52
Acts of Violence	Q56 – Q59
U.S. Actions	Q60, Q62
Demographics	D12 – D17, D21 – D24, D26

Table 2. Related questions specific to the analysis

Crucial to any quantitative modeling of survey data is the appropriate preparation of the data. The first step in this process is re-coding the responses from the original Likert scale responses to numeric values. Various Likert scales were used in the survey and they differed both in terms of qualitative scales and response ranges. For example, a four-point Likert scale accounted for 66% of the total number of questions. Typically the response scales were in the form of “always positive,” “somewhat positive,” “somewhat negative” and “always negative,” or some other similar positive to negative range. The survey also had questions with five-point Likert scale responses as well as binary responses. Recoding was done using the *CAR* package with the R statistical software program. Before the data was re-coded, it was important to determine how the response would be viewed. The factors used in this analysis were re-coded in a positive or negative direction depending on how a U.S. analyst would interpret the numeric variables loaded onto the factors. Consistency in the direction of the recoded variables does ease the burden of interpretation once the factors have been defined and the linear models fit. In general if a response was assumed to be positive to a U.S. analyst, then the response was given a positive value, and if it was assumed to be negative, then it was given a negative value. Numeric re-coding values range from a +2 to a -2. If the range was a four-point Likert scale, then the extreme positive and negative answers were given a +2 and -2 respectively. The moderate positive and negative were given a +1 and -1 respectively. The re-coding values for a fivepoint Likert scale is similar to a four point one, but with a 0 coded for neutral type responses such as “stayed the same.” Three-point Likert scales have a +2 and -2 for extremes and 0 coded to neutral responses, but there are no moderate values. There were also questions that offered binary responses, such as a general “Oppose” or “Support,” and a more formal choice of response as “Shari’ah reduces crime in society” or “Shari’ah does not affect the amount of crime in society.” These types of questions were given values of -2 or 2 (Kulzy, 2012).

The “Don’t know” and “No response” responses in this data were treated as unknown values that needed to be imputed. This is in contrast to the typical solution for handling missing data, which is to remove the associated entire observation from the data. This approach is often referred to as casewise deletion. In terms of survey analysis, casewise deletion means that if a respondent failed to respond to one question, then all of the rest of his or her information from the other 141 questions would be removed. For this data set, if casewise deletion was used in order to be able to first conduct a factor analysis and subsequently fit regression models, 2,240 of the 3,770 Nigerian observations (60%) would be removed from analysis. This is in spite of the fact that each question only had a very small percent of missing responses. Thus, imputation is crucial to this survey because imputing only 6% -8% of the data saves 60-72% of it for analysis. Missing data was handled using nearest neighbor hot-deck imputation, a more sophisticated method than simple mean imputation, and was implemented using the state or region as a matching variable in order to account for spatial variation in the data (Kulzy, 2012). The hot-deck imputation method used in this effort is based on the *RANDwNND.hotdeck* function within *StatMatch* package of R. The imputation for the missing values, to include “Don’t knows” and “No Response” responses, was done using the variables: region/state (the states of the country), “d5a” (religion), “d0” (gender), “urban/rural” (live in urban or rural area of state). The *RANDwNND.hotdeck* function initially subsets the data based on specific “donor class” variables. For this research, the donor class variable is the “state” variable. Basing the donor class on geographic state ensures that geographic heterogeneity is accounted for in the imputation. Within each state, then, the data is subset into two groups: the receivers and the donors. The receivers are those respondents who are missing the response to a particular question and the donors are those respondents who have answered the same particular question. For each receiver, a donor is then identified that is closest to the receiver in terms of Manhattan distance based on his or her religion, gender, and location (urban/rural). If there is more than one “closest” donor, then ”one is picked at random” from among the tied group of the closest matches (D’Orazio, 2011).

Imputing all of the ”Don’t know” responses could have an impact on a few questions that loaded onto a factor with a minimal significant value of 0.4. Those questions loading as a 0.4 in one imputation would be considered significant. However, if the process was to be repeated, there is a chance that a minimal, loaded value question may now fall below the 0.4 threshold and be removed from the factor. It was determined to recode the ”Don’t know” responses in a manner that minimized the volatility of these few questions which rest on the cusp of the 0.4 threshold. It was assumed that a “Don’t know” in the three and five point Likert scales would be equivalent to a “No Response” because a neutral, valued at zero, response was offered. Therefore, three and five point Likert scales of “Don’t know” were imputed in the same manner as a “No Response.” A more difficult question is how to best analyze “Don’t know” responses in a two- and four-point Likert scales since these types of scales do not offer an explicit neutral response option. It is reasonable to assume that a “Don’t know” response to a question with only “Strongly agree”, “Agree,” “Disagree,” and “Strongly disagree” could, in fact, be using the “Don’t know” response option to express neutrality, particularly when there was also a “No response” option. Thus, in these cases a “Don’t know” response was re-coded to a value of 0, a choice which seems conservative in the sense that without it imputing these responses would result in a potentially neutral person being given a positive or negative response (Kulzy, 2012). This assumption addresses over 60% of the missing data that would have otherwise required imputation. Roughly 6-8% of Nigeria’s questions did not have a clear response, and eight of these questions are asked on either a two- or four-point Likert scale for Nigeria. Since there is no clear and definitive interpretation of the “Don’t know” responses for these questions, and because of the large number of these questions, a closer analysis was performed. It is plausible to believe that without an option to be neutral, as in two- and four-point Likert scales, a logical interpretation of “Don’t know” is neutral which would then result in re-coding it to zero. If this were to be the case then these questions would not be explicitly imputed. However, this is not necessarily true for other types of questions (Kulzy, 2012).

4.4 FACTOR ANALYSIS

One of the major challenges with large surveys is reducing the mass of data into useful information. Another challenge with surveys aimed at understanding the human terrain, particularly when applied to irregular warfare, is that the population characteristics of interest may not be directly measured via single questions. Factor analysis helps address both of these issues.

Critics of the factor analysis argue that its inherent subjectivity and flexibility allows analysts to manipulate the output. The non-unique solution of the factor loadings is often particular focus of this criticism. However, all mathematical and statistical models can be manipulated, and most involve making numerous subjective choices (choice of variables, model parameterization, etc). In this sense, factor analysis is no different. As with those methods, and research in general, it is incumbent on the researcher to ensure his or her results are not sensitive to, or dependent on, modeling choices. That said, remember that the goal of factor analysis is to create factors that are both statistically and substantively meaningful, and the latter implies -- perhaps requires -- a degree of subjectivity.

Factor analysis is a hybrid of social and statistical science. First conceived in the early 1900s, the goal was multivariate data reduction, but data reduction of a very specific type.

Essentially the idea is to explain the correlation structure observed in *p* dimensions via a linear combination of *r* factors, where the number of factors is smaller than the number of observed variables, and where the factors achieve both "statistical simplicity and scientific meaningfulness" (Harman, 1976).

Figure 2 illustrates the idea of factor analysis with six observed variables (i.e., survey question responses) that can be effectively summarized in terms of two latent variables (factors). Note that the survey question responses are observed with error (denoted by the ε_i terms) and the question responses are weighted linear combinations of the factors (where the weights are the λ_{ij}s). What factor analysis does is model the *p* observed variables as linear combinations of *r* factors, where the analyst has to pre-specify *r*, such that the model covariance matrix closely matches the sample covariance matrix of the observed variables.

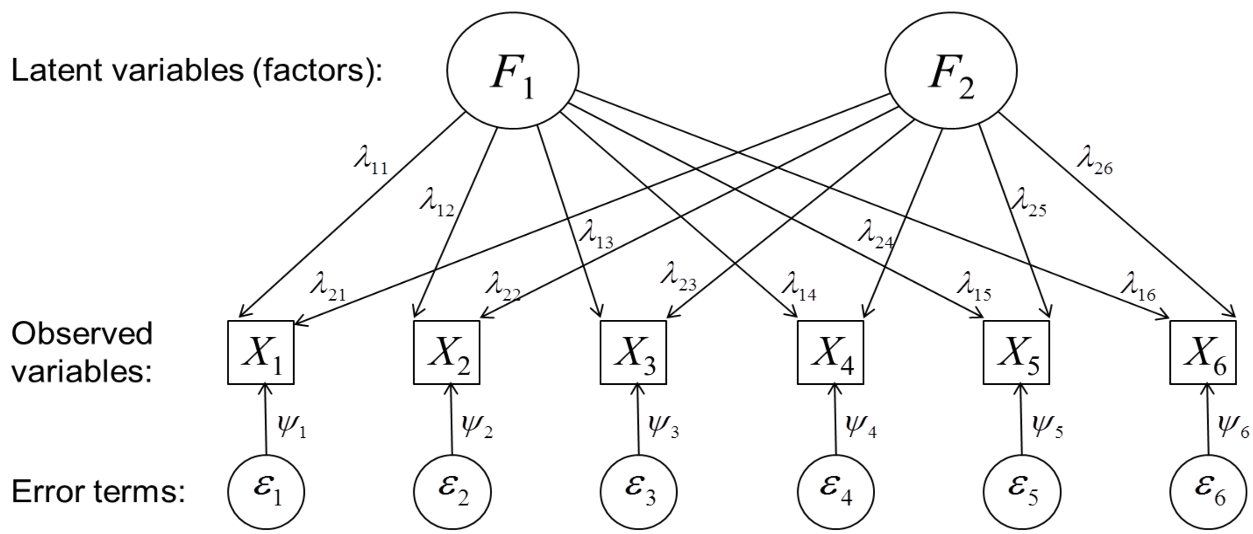

Figure 2. An illustrative example of factor analysis with six observed variables that can be effectively summarized in terms of two latent variables (factors).

An alternative to factor analysis is principal components which uses orthogonal transformations to convert a set of possibly correlated variables into a reduced set of uncorrelated variables that capture most of the variation in the original data. The transformation is defined so that the first principal component accounts for as much of the variability in the data as possible, and each succeeding component has the highest variance possible under the constraint that it be orthogonal to the preceding component or components. A principal components analysis, while useful for efficiently summarizing data, does not necessarily result in factors with scientifically meaningful interpretations.

In contrast, factor analysis is specifically designed to look for meaningful commonality in a set of variables (DeCoster, 1998). There are two types of factor analysis: exploratory factor analysis (EFA) and confirmatory factor analysis (CFA). EFA looks to explore the data to find an acceptable set of factors. In this sense, it is much like exploratory data analysis. The goal is not so much to formally test hypotheses as it is to discover likely factors that will account for at least 50 percent of the common variation in the observed factors. CFA, on the other hand, begins with a theory or hypothesis about how the factors should be constructed and seeks to test whether the hypothesized structure adequately fits the observed data.

4.4.1 The Factor Analysis Model

Consider a survey consisting of *p* questions given to *n* respondents, where respondent *i*'s responses are denoted $\boldsymbol{y}_i = \{y_{i1}, \dots, y_{ip}\}$. From the data, a sample covariance matrix **S** is calculated in the usual way for the set of centered variables,

$$\boldsymbol{x}_i \triangleq \{y_{i1} - \bar{y}_1, \dots, y_{ip} - \bar{y}_p\},$$

where $\bar{y}_j = \frac{1}{n}\sum_{i=1}^{n} y_{ij}$. That is, the $j(k)^{\text{th}}$ entry of **S** is calculated as $s_{jk} = \frac{1}{n-1}\sum_{i=1}^{n} x_{ij}x_{ik}$, $j \in \{1,2, \dots, p\}$ and $k \in \{1,2, \dots, p\}$.

The fundamental assumption of factor analysis is that, for some $r < p$, each of the *p* centered variables ($\boldsymbol{X} = \{X_1, \dots, X_p\}$) can be expressed as the sum of *r* common factors ($\boldsymbol{F} = \{F_1, \dots, F_r\}$) multiplied by their loadings $(\lambda_{i1}, \dots, \lambda_{ir})$ plus a unique factor ($\boldsymbol{E} = \{\varepsilon_1, \dots, \varepsilon_p\}$) multiplied by its associated loading $(\psi_1, \dots, \psi_p)$,

$$\begin{aligned}
X_1 &\triangleq Y_1 - \mu_1 = \lambda_{11}F_1 + \lambda_{12}F_2 + \cdots + \lambda_{1r}F_r + \psi_1\varepsilon_1 \\
X_2 &\triangleq Y_2 - \mu_2 = \lambda_{21}F_1 + \lambda_{22}F_2 + \cdots + \lambda_{2r}F_r + \psi_2\varepsilon_2 \\
&\vdots \\
X_p &\triangleq Y_p - \mu_p = \lambda_{p1}F_1 + \lambda_{p2}F_2 + \cdots + \lambda_{pr}F_r + \psi_p\varepsilon_p \qquad (1)
\end{aligned}$$

where $\mu_j = \mathbb{E}(Y_j)$. Now, while the above formulation looks similar in many respects to a series of linear models, note that *everything* on the right-hand side of the *p* equations is *unobserved*. In spite of that, the goal is to estimate the loadings from the data so that the modeled covariance matrix **R** is "close to" the observed sample covariance matrix **S**.

Using matrix notation, Equation (1) can be expressed compactly as

$$\boldsymbol{X} = \boldsymbol{\Lambda F} + \boldsymbol{\Psi E}, \tag{2}$$

where $\mathbf{\Lambda}$ is the matrix of the loadings for the common factors of dimension $p \times r$ and $\mathbf{\Psi}$ is a matrix of dimension $p \times p$ with $\psi_1, \ldots, \psi_p$ on the diagonal and all off diagonal entries zero. Assuming $\mathbb{E}(\boldsymbol{E}) = \mathbf{0}$, we get to the whole point in fitting the factor analysis model, which is that we can use the estimated common factor loadings $\widehat{\boldsymbol{\Lambda}}$ to express the factors in terms of their constituent parts:

$$\mathbb{E}(\boldsymbol{F}) = \widehat{\boldsymbol{\Lambda}}^{-1}\mathbb{E}(\boldsymbol{X}). \tag{3}$$

One of the most common uses of exploratory factor analysis is to "determine what sets of items hang together in a questionnaire" (DeCoster, 1998). Thus, assuming Equation 1 is an appropriate model, via Equation 3 we can determine which of the survey questions are most related and, as desired, use them to estimate the underlying latent factor for any respondent as a linear combination of their responses to the survey questions. Furthermore, if the scientific meaningfulness goal is achieved, the latent variables will have useful and interpretable meanings that provide additional insight into the characteristics of the populations being studied.

Of course, at this point it should be evident that there will be no unique solution to this problem. There are simply too many degrees of freedom in the problem formulation and, even after some assumptions to make the problem solvable, there will still be an infinite set of solutions. This, along with the fact that the choice of solution is subjective, is one of the frequent criticisms of factor analysis. Nonetheless, as we will show, we have found the results to be quite informative and useful in our survey analyses, and there are ways to minimize the number of subjective modeling choices that must be made. There are three critical steps in fitting a factor analysis model: (1) Determining the number of factors, (2) fitting the model in order to estimate the common factor loadings, and (3) rotating the loadings to find the preferred solution. We discuss each of these in turn.

4.4.2 Determining the Number of Factors

To conduct factor analysis, one must pre-specify the number of factors r to fit. In so doing, it is crucial not to underestimate or overestimate the number of factors. If too few factors are chosen then the fitted factors become overloaded with irrelevant variables. On the other hand, with an excessive number factors the variables may be spread out too much over the fitted factors. In either case, the result is likely to be that meaningful factors are never properly revealed.

This seems like a catch-22: To determine the correct factors, one must first know how many factors there are. However, over the years a number of solutions have been proposed, some that work better than others.

One early solution is the Kaiser rule which stipulates that the number of factors used in the model should equal the number of eigenvalues for the original data matrix that are greater than one. Another is to use a Scree plot to graph successive eigenvalues versus the number of factors and then setting r to the number of factors where the plotted line visually levels out (indicating that the remaining factors have little explanatory power).

The difficulty with the Kaiser rule and the Scree plot is they are heuristics. The Kaiser rule was designed to help the analyst of the early- to mid-1900s get “into the ballpark” with respect to an acceptable number of factors, but then the analyst was supposed to further refine the acceptable number of factors through trial and error. The Scree plot is also a heuristic because it allows for subjectivity in interpreting the plotted line where, to determine the number of factors, the analyst must visually determine when the line in the Scree plot levels out.

An alternative to these methods, which only became feasible with the widespread availability of significant computing power, is parallel analysis. Parallel analysis involves the construction of multiple correlation matrices from simulated data, where the average eigenvalues from the simulated correlation matrices are then compared to the eigenvalues from the real data correlation matrix. The idea of parallel analysis is that factors derived from the real data should have larger eigenvalues than equivalent factors derived from repeatedly resampled or simulated data of the same sample size and number of variables. Then *r* is set to the number of factors in the actual data that are greater than the average of the equivalent simulated data factor eigenvalues (Hayton, Allen, & Scarpello, 2004).

4.4.3 Fitting the Model

Given that by definition $\mathbb{E}(\boldsymbol{X}) = \mathbf{0}$, and assuming that the common factors are independent of the unique factors, it is straightforward to show that the covariance matrix for **X** from Equation 2 is

$$\boldsymbol{R} = \boldsymbol{\Lambda}\boldsymbol{R_F}\boldsymbol{\Lambda}' + \boldsymbol{\Psi}^2, \tag{4}$$

where $\mathbf{R}_F$ is the covariance matrix of the factors (Mulaik, 2009). Further assuming that $\mathbb{E}(\boldsymbol{F}) = 0$ and $cov(\boldsymbol{X}) = \boldsymbol{I}$, where the former condition follows because the factors can always be rescaled and the latter because we assume the factors are independent, Equation 4 simplifies to

$$\boldsymbol{R} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}' + \boldsymbol{\Psi}^2. \tag{5}$$

Then from Equation 5, $\mathbf{\Lambda}$ and $\boldsymbol{\Psi}$ are estimated via maximum likelihood.

Note that the maximum likelihood estimators (MLEs) are not analytically derivable and must be solved for numerically using an iterative approach. Under the assumption that **F** and **E** are jointly normally distributed, the calculations essentially follow the usual estimation methods with an additional uniqueness condition added because of the indeterminacy of the factor analysis model.

4.4.4 Choosing the Preferred Rotation

Maximum likelihood estimation results in a non-unique solution for how the variables load onto the factors. That is, for any estimated common factor loading matrix $\widehat{\boldsymbol{\Lambda}}$ there are infinitely many other matrices that will fit the observed sample covariance matrix **S** equally well since

$$\widehat{\boldsymbol{\Lambda}}\boldsymbol{F} = \widehat{\boldsymbol{\Lambda}}\boldsymbol{T}\boldsymbol{T}^{-1}\boldsymbol{F} = \boldsymbol{\Lambda}^*\boldsymbol{F}^*, \tag{6}$$

where $\boldsymbol{\Lambda}^* = \widehat{\boldsymbol{\Lambda}}\boldsymbol{T}$ and $\boldsymbol{F}^* = \boldsymbol{T}^{-1}\boldsymbol{F}$ for some transformation matrix **T**.

Thus, after an initial solution is found, the final step in factor analysis is to rotate the variables to simplify their factor loadings. The rotation process is critical to factor analysis because it allows the analyst to identify the desired factor constructs, usually in terms of a simple structure of substantively interesting variables. However, this procedure is susceptible to criticism because all rotations are mathematically equivalent and thus the final choice is subjective.

There are two main types of rotation: (1) oblique, and (2) orthogonal. Orthogonal rotation is most commonly associated with what is called the "varimax" method, and oblique rotations are most commonly associated with what is called the "promax" method. The distinction between the two rotations is whether the factors are assumed to be correlated or not; orthogonal rotations are uncorrelated while oblique rotations may be correlated.

Kline says the most accepted method for creating factors with simple structure is varimax (Kline, 1994). On the other hand, the oblique method is recommended by Costello & Osborne because it can account for both correlated and uncorrelated factors (Costello & Osborne, 2005).

We used the varimax rotation on our survey data and found it to work well. As defined in Johnson & Wichern, the varimax procedure finds an orthogonal transformation matrix **T** that maximizes

$$V = \sum_{j=1}^{r} \left[\sum_{i=1}^{p} \tilde{\lambda}_{ij}^4 - \frac{1}{p} \left(\sum_{i=1}^{p} \tilde{\lambda}_{ij}^2 \right)^2 \right], \tag{7}$$

where $\tilde{\lambda}_{ij} = \hat{\lambda}_{ij} / \sqrt{\sum_{j=1}^{r} \hat{\lambda}_{ij}^2}$ (Johnson & Wichern, 2002). Equation 7 is akin to calculating the sum of the variances of the factor loadings across the r factors. What varimax does is find the rotation that makes the high loadings as high as possible while simultaneously making the low loadings as low as possible on each factor.

4.4.5 Factor Analysis of the 2010 Nigeria Survey Data

As mentioned in Section 4.2, the Nigeria survey was fielded in 2010 to 3,770 respondents. A sample of sufficient size is an important consideration since the sample covariance matrix **S** is an estimate of some underlying true covariance matrix **Σ**. That is, since factor analysis focuses only on the sample covariance matrix, it is important that **S** is in fact a good estimate of $\mathbf{\Sigma}$ in order to ensure the resulting factors represent underlying features of the population and not the noise or other artifacts of the sample.

The factor analysis models were fit using the R statistical package. In particular, the *factanal* function in the base package was used to fit the factor analysis model and rotate the loadings to get the final solution. And, we used the *fa.parallel* of the R *psych* package to do the parallel analyses (Revelle, 2011).

Prior to fitting the factor analysis models, we first cleaned and coded the data, and then we imputed a small number of missing values in order to prepare the data as described previously in detail in Section 4.3. The most important point to make here is that factor analysis can only be done with complete data and thus imputation is a critical step to complete prior to doing factor analysis. For our data, approximately six percent of the data was missing (due, for example, to respondents refusing or failing to answer one or more questions), but they were spread throughout the data set. Thus, if we had only used complete records, we would have eliminated 60 percent of the respondents. Imputation allowed us to use all the data and subsequent sensitivity analyses demonstrated that our imputation assumptions had no practical effect on the factor analysis results.

Returning to factor analysis, as discussed in Section 4.4.2, we first used parallel analysis to determine r, the number of factors. Figure 3 shows the results from *fa.parallel* for Nigeria, where the eigenvalues for 27 factors were greater than those from the simulated data (the blue line is greater than the dashed red line), so we set $r = 27$. Sensitivity analysis using other values of r subsequently confirmed that $r = 27$ was indeed appropriate. In the end, however, we only used 26 factors, as the last one contained low factor loadings, contained only two questions that were also repeated in another factor, and was therefore not used in this analysis. Of note, also is the fact that for this research, variables with loadings between 0.4 and -0.4 were removed.

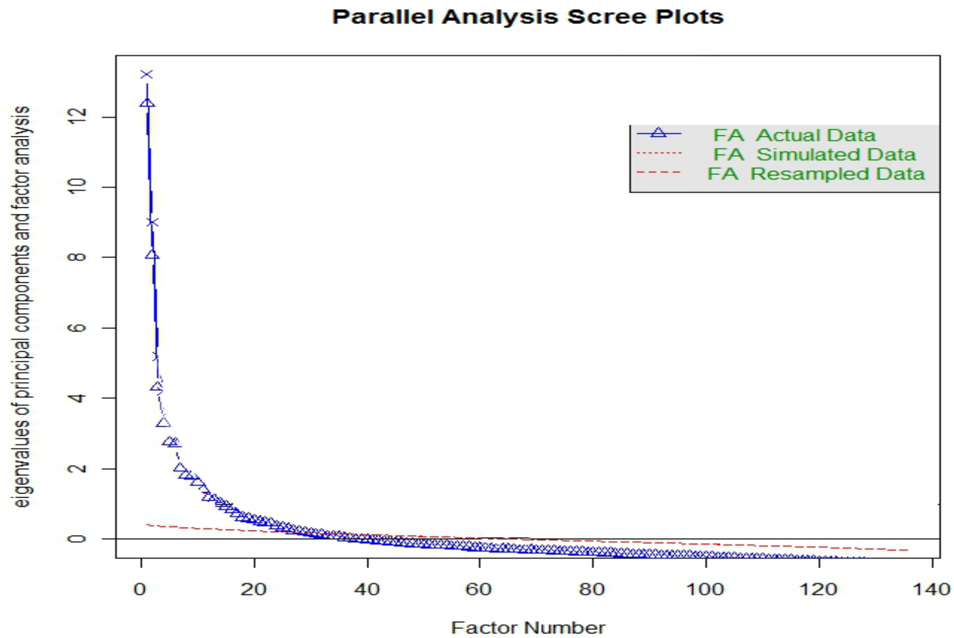

Figure 3. Parallel analysis where the eigenvalues for 27 factors were greater than those from the simulated data (the blue line is greater than the dashed red line).

The list of the factors and the questions that load onto each is given below in Figure 4. Each factor name was chosen subjectively based on the content of the questions that loaded onto each particular factor. These 26 factors, in addition to 4 other survey questions that were not used in the factor analysis, will become the variables used in the next part of the project where we build regression models in order to predict future population issue stance scores and observed attitudes and behaviors.

The final step in this factor analysis is to compute a factor score for each respondent. This is a necessary step if we wish to conduct further analysis with the factors or to use them in any kind of model building. The score for a given factor is simply the linear combination of each measure or question, weighted by the corresponding factor loading (DeCoster, 1998). We can further refine this by rescaling the resulting factor score by dividing by the column (factor score) sums, thereby obtaining a factor score of between -2 and 2, the same as our recoded scale as described in Section 4.3.

Factor Name	Factor No.	Questions							
Shari'a Law	X1	q32a	q32b	q32c	q32d	q32e	q33	q57	
U.S. Assistance to Nigeria	X2	q21a	q21b	q21c	q21d	q21e	q21f	q21g	q21h
Chinese Assistance to Nigeria	X3	q22a	q22b	q22c	q22d	q22e	q22g	q22h	
Social & Essential Services	X4	q8edu	q8hea	q8wat	q9edu	q9hea	q9wat		
Trust in Government Agencies	X5	q49na	q49pp	q49af	q49cj	q49lp	q49lg		
External Security	X6	q23b	q23c	q23d	q23e	q23f			
General Trust	X7	q26a	q26b	q26c	q26d	q26e			
Non-Western Countries	X8	q12ni	q12ir	q16so	q16li	q16sa			
Local & National Freedom	X9	q48a	q48b	q48c	q48d	q48e	q48f		
Democracy	X10	q40	q42	q44	q45				
Other's Values	X11	q17sa	q17fr	q17ch	q17ir	q17us			
Daily Life Acceptance	X12	q27a	q27b	q29a	q29b				
Use of Violence	X13	q25a	q25b	q25c					
Terrorism Enablers	X14	q23a	q59d						
Family & Friends	X15	q27c	q27d	q29c	q29d				
Civic Duty	X16	d24a	d24b						
Attacks on U.S.	X17	q58a	q58b	q58c					
Discussion of U.S.	X18	q62a	q62b	q62c					
Electricity	X19	q8ele	q9ele						
Western Countries	X20	q12uk	q12fr	q14usa					
Trust in Policy Makers	X21	q49pr	q49pm	q50					
Religious Freedom in the West	X22	q37c	q37d						
Religious Intolerance	X23	q36a	q36b						
Civility	X24	q28	q30						
Policy and Law	X25	q31a	q31b						
Roads	X26	q8roa	q9roa						

Figure 4. List of factors and factor names

4.5 PREDICTIVE MODELS

We now move on to use what we have done with the data through the recoding, imputation, and factor analysis to building regression models that will enable us to predict a population's response in light of future events within the context of the TRAC IW TWG.

In short, the IW TWG seeks to stimulate a player such that he/she are forced to make the "best" decisions and develop appropriate courses of action in a given location and scenario. In order to do this, the game model must be able to provide feedback from the local populace to the player on how player decisions effect population perceptions. The subsequent linear and multinomial logistic regression models that predict population responses were built specifically with this functionality in mind, to stimulate player action and decision making in a simpler, and more traceable way than is currently being used with TRAC's Cultural Geography model.

4.5.1 Predicting Issue Stance Scores Using Linear Regression

The first step in building linear regression models used to predict future issue stance scores and the subsequent OABs (though using different model), is to determine what issues are most important to the population. That is, of all of the factors that we identified during the factor analysis, which ones matter most to the people as well as providing the most predictive power? To do this, we take the 26 factors and 4 other survey questions (q6, q7, q10, d23) that were not used in the factor analysis (this will avoid multi-colinearity problems), and regress each against all the other ones, thereby creating 30 linear regression models all with 29 predictor variables (no interaction terms were used). In order to create the simplest predictive model that minimizes over-fitting, we use a stepwise deletion process, specifically the *stepAIC* function in R. This function, in order to find the statistically significant predictor variables, deletes the term with the highest *p*-value (greater than 0.05), re-runs the model, and continues this process until all the remaining variables have *p*-values that are less than 0.05. The 30 models, now simplified with only significant predictor terms remaining, are then compared based on their adjusted R^2 value. Those models with an adjusted R^2 of greater than 0.4, and that do not violate any of the usual linear regression modeling assumptions, are chosen as the "best" ones, and in this context represent the key issues that matter most to the population as well as those with the most predictive influence. Each of the four factors X2, X4, X5, and X10, also account for a large proportion of the total variance, again indicating that these four are the key issues to the population. We get four that meet these criteria: models with X2, "U.S. Assistance to Nigeria", X4, "Social & Essential Services", X5, "Trust in Government Agencies", and X10, "Democracy", as the response variables. Since we don't want any one of the four response variables being predictor variables in one of the other four's regression equation, we re-build each of the four models, taking out the other three response variables if they were present as predictors. Our four issue stance / linear regression equations are then given by:

$$X_2 = -0.19 + 0.03X_1 + 0.38X_3 + 0.07X_6 - 0.08X_8 + 0.05X_9 + 0.09X_{14} + 0.03X_{17} + 0.12X_{18} + 0.24X_{20} + 0.08X_{21} + 0.03X_{22} + 0.02X_{23} + 0.03X_{26} + 0.05q_7$$

$$X_4 = 0.09 + 0.09X_3 - 0.04X_6 - 0.05X_{13} + 0.09X_{14} - 0.04X_{15} - 0.01X_{16} - 0.02X_{17} + 0.11X_{19} + 0.04X_{20} + 0.04X_{21} + 0.04X_{22} - 0.02X_{23} + 0.12X_{24} + 0.05X_{25} + 0.3X_{26} + 0.05d_{23} + 0.09q_6 + 0.13q_7 + 0.02q_{10}$$

$$X_5 = -0.51 - 0.03X_1 + 0.1X_3 + 0.16X_7 + 0.03X_8 + 0.16X_9 + 0.02X_{11} + 0.08X_{12} - 0.04X_{14} - 0.02X_{15} + 0.02X_{16} + 0.05X_{18} + 0.06X_{19} - 0.03X_{20} + 0.35X_{21} - 0.02X_{22} - 0.04X_{23} - 0.03X_{25} + 0.05X_{26} - 0.02q_6 + 0.07q_7 + 0.02q_{10}$$

$$X_{10} = -0.06 + 0.03X_3 + 0.09X_7 - 0.13X_8 + 0.18X_9 + 0.1X_{11} - 0.06X_{13} - 0.04X_{14} + 0.04X_{15} + 0.05X_{16} - 0.03X_{20} + 0.31X_{21} + 0.02X_{22} - 0.03X_{23} + 0.05X_{24} + 0.05X_{26} + 0.05d_{23} + 0.02q_6 + 0.2q_7 + 0.05q_{10}$$

These regression equations will now allow us to predict future issue stance scores, which will be demonstrated through a small use case in Section 4.5.3.

4.5.2 Predicting Future OABs Using Multinomial Logistic Regression

In the previous section, we showed how a linear regression model can be used to predict future issue stance scores from a given population. We now move on to the next step, predicting future observed attitudes and behaviors (OABs) using a different type a model, the multinomial logistic regression.

A simple logistic regression model can be used in situations where the response variable is dichotomous or binary, that is, the response measurement for each subject is a "success" or "failure". This model type can be modified to handle cases where the outcome variable is nominal with more than two levels (Hosmer & Lemeshow, 2000). For instance, we could employ a multinomial logistic regression if we wanted to model the choice of a meal plan from among three offered to students at a university. If the meal plans are represented by "A", "B", and "C", we could model, based on whatever predictor variables we have chosen, the probability of a student choosing one of the three meal plans as a function of those covariates. We must, however, pay attention to the scale that is used, as different methods can be employed if the scale is nominal or ordinal (Hosmer & Lemeshow, 2000). For our purposes here, we will use a nominal scale. To develop the model, assume we have p covariates and a constant term, denoted by the vector **x**, of length $p + 1$. Since we have three outcome variables in our meal plan example, we will need two logit functions, and we will compare the baseline outcome, meal plan "A" (or $P(Y = 0)$), against the others. We denote the two logit functions as:

$$g_1(\mathbf{x}) = ln\left[\frac{P(Y=1|\mathbf{x})}{P(Y=0|\mathbf{x})}\right] = \beta_{10} + \beta_{11}x_1 + \beta_{12}x_2 + \cdots + \beta_{1p}x_p \text{ , and}$$

$$g_2(\mathbf{x}) = ln\left[\frac{P(Y=2|\mathbf{x})}{P(Y=0|\mathbf{x})}\right] = \beta_{20} + \beta_{21}x_1 + \beta_{22}x_2 + \cdots + \beta_{2p}x_p \text{ .}$$

Notice that there are separate parameters for each logit function, meaning that the effects vary according to the response category paired with the baseline (Agresti, 1996). The conditional probabilities of each of the three outcome variables given **x** are then:

$$P(Y=0|\mathbf{x}) = \frac{1}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})}} \text{ ,}$$

$$P(Y=1|\mathbf{x}) = \frac{e^{g_1(\mathbf{x})}}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})}} \text{ , and}$$

$$P(Y=2|\mathbf{x}) = \frac{e^{g_2(\mathbf{x})}}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})}} \text{ .}$$

A general expression for the conditional probability in an *n* category model is:

$$P(Y=j|\mathbf{x}) = \frac{e^{g_j(\mathbf{x})}}{\sum_{k=0}^{n-1} e^{g_k(\mathbf{x})}} \text{ .}$$

We can estimate the value of the parameters by first constructing a likelihood function for a sample of *n* independent observations, given by:

$$l(\boldsymbol{\beta}) = \prod_{i=1}^{n} [\pi_0(\boldsymbol{x}_i)^{y_{0i}} \pi_1(\boldsymbol{x}_i)^{y_{1i}} \pi_2(\boldsymbol{x}_i)^{y_{2i}}] \text{ .}$$

By taking the log of this likelihood function we get:

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{n} y_{1i} g_1(\boldsymbol{x}_i) + y_{2i} g_2(\boldsymbol{x}_i) - ln\left(1 + e^{g_1(x_i)} + e^{g_2(x_i)}\right) \text{ .}$$

The likelihood equations are constructed by taking the first partial derivatives of $L(\boldsymbol{\beta})$ with respect to each of the unknown parameters. The general form of these equations is:

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}_{jk}} = \sum_{i=1}^{n} \boldsymbol{x}_{ki}\left(y_{ji} - \pi_{ji}\right).$$

The maximum likelihood estimator is then obtained by setting these likelihood equations equal to zero and solving. The solution requires the same type of iterative computation that is used in the simpler binary outcome case (Hosmer & Lemeshow, 2000). For a more detailed discussion, see *Applied Logistic Regression* by Hosmer & Lemeshow.

With now a basic understanding of the multinomial logistic regression model, we can move on to a description of the methodology that we used in order to predict future OAB scores. The goal here is to determine with what probability, after a game event occurs, the population will blame an actor for that event happening, and to see over time with a small use case that follows from section 4.5.1, how these probabilities change. As our response variable, we used question 47 of the survey described earlier in section 4.2. The question asked: "In your opinion, which of the following groups is most to blame for ongoing violence in your country today?" The response options were: "Rebel Groups", "International Terrorists", "Common Criminals", "The Military", "Government Officials", or "Foreign Countries". This particular question was chosen because it was the only one that asked about the specific actors that we felt were most relevant in an IW TWG scenario. Since we wanted a samples' issue stance score to have some influence over their OAB towards an actor, we built a multinomial logistic regression model with question 47 as the six category response variable, and the four key issues, X2, X4, X5, and X10, as the predictor variables. The *mlogit* library in the R statistical package gives us the following five logit functions, using "Rebel Groups" as the baseline, where $\mathbf{x} = \langle X_2, X_4, X_5, X_{10} \rangle$:

$$g_1(\mathbf{x}) = ln\left[\frac{P(Y = Terrorists|\mathbf{x})}{P(Y = Rebels|\mathbf{x})}\right] = -0.51 - 0.31X_2 + 0.06X_4 + 0.01X_5 + 0.23X_{10}\ ,$$

$$g_2(\mathbf{x}) = ln\left[\frac{P(Y = Criminals|\mathbf{x})}{P(Y = Rebels|\mathbf{x})}\right] = 0.85 - 0.34X_2 - 0.01X_4 + 0.11X_5 + 0.08X_{10}\ ,$$

$$g_3(\mathbf{x}) = ln\left[\frac{P(Y = Military|\mathbf{x})}{P(Y = Rebels|\mathbf{x})}\right] = -0.21 - 0.12X_2 + 0.06X_4 - 0.09X_5 + 0.04X_{10}\ ,$$

$$g_4(\mathbf{x}) = ln\left[\frac{P(Y = Government|\mathbf{x})}{P(Y = Rebels|\mathbf{x})}\right] = 1.8 - 0.14X_2 + 0.02X_4 - 0.16X_5 - 0.2X_{10}\ ,\text{and}$$

$$g_5(\mathbf{x}) = ln\left[\frac{P(Y = Foreign|\mathbf{x})}{P(Y = Rebels|\mathbf{x})}\right] = -1.2 - 0.07X_2 - 0.02X_4 - 0.01X_5 - 0.3X_{10}\ .$$

The six conditional probability models are then given as:

$$P(Y = Rebels|\mathbf{x}) = \frac{1}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})} + e^{g_3(\mathbf{x})} + e^{g_4(\mathbf{x})} + e^{g_5(\mathbf{x})}},$$

$$P(Y = Terrorists|\mathbf{x}) = \frac{e^{g_1(\mathbf{x})}}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})} + e^{g_3(\mathbf{x})} + e^{g_4(\mathbf{x})} + e^{g_5(\mathbf{x})}},$$

$$P(Y = Criminals|\mathbf{x}) = \frac{e^{g_2(\mathbf{x})}}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})} + e^{g_3(\mathbf{x})} + e^{g_4(\mathbf{x})} + e^{g_5(\mathbf{x})}},$$

$$P(Y = Military|\mathbf{x}) = \frac{e^{g_3(\mathbf{x})}}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})} + e^{g_3(\mathbf{x})} + e^{g_4(\mathbf{x})} + e^{g_5(\mathbf{x})}},$$

$$P(Y = Government|\mathbf{x}) = \frac{e^{g_4(\mathbf{x})}}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})} + e^{g_3(\mathbf{x})} + e^{g_4(\mathbf{x})} + e^{g_5(\mathbf{x})}}, \text{and}$$

$$P(Y = Foreign|\mathbf{x}) = \frac{e^{g_5(\mathbf{x})}}{1 + e^{g_1(\mathbf{x})} + e^{g_2(\mathbf{x})} + e^{g_3(\mathbf{x})} + e^{g_4(\mathbf{x})} + e^{g_5(\mathbf{x})}}.$$

These multinomial logistic regression equations will be used in our use case to determine future observed attitudes and behaviors of the population towards each actor in the proof of principle scenario that follows. In order to determine how well our models fit the data, we could subset our data into a training set as well as a test set, re-build our models on the training set, apply these to our test set, and see how well our models predict our response variable. Ideally, our test set would be next year's survey, assuming of course the same questions are asked, enabling us to determine the predictive power of our models.

4.5.3 Proof of Principle Scenario

In order to predict future issue stance scores, we would require a certain amount of subject matter expert (SME) input. That is, for each event scheduled to happen during our small use case, we would need to solicit SME input in order to determine how these would affect population views with respect to the 26 factors and 4 additional survey questions. Each of the 30 variables would get a score between -2 and 2 for each event, with -2 corresponding to a highly negative impact, -1 to a slightly negative impact, 0 to no impact, 1 to a slightly positive impact, and 2 to a highly positive impact. For our purposes in this project, as it is only a "proof of principle", SME input was notional and generated in a random fashion using an Excel spreadsheet and input into the models from there (see Appendix D). Additionally, if we should use this methodology during an actual IW TWG, we would probably want to subset the data into different population stereotypes before building our models, and then use those models and SME input as described above for each separate stereotype. This would enable us to more effectively model the population. But again, as this was only a “proof of principle”, we built one set of models for the entire population. We first need to calculate the initial issue stance score and OAB probabilities in order to instantiate our model. The initial issue stance score will result in a number between -2 and 2 (the same range as the re-scaled factor scores), and is accomplished by using the mean score for each factor as input for each of the four separate equations. The initial issue stance scores are given in Table 3.

Response Variable	Initial Issue Stance Score
X2. U.S. Assistance to Nigeria	0.178
X4. Social & Essential Services	0.151
X5. Trust in Government Agencies	-0.145
X10. Democracy	0.272

Table 3. Initial issue stance ccores by key issue

The initial OABs are calculated similarly, using the mean factor scores for X2, X4, X5, and X10 as inputs for our conditional six probability models. The initial OAB probabilities are given in Table 4.

Actor	Initial OAB Probability
Rebel Groups	0.093
International Terrorists	0.057
Common Criminals	0.206
Military	0.076
Government Officials	0.541
Foreign Countries	0.027

Table 4. Initial OAB probabilities by actor

Once our initial issue stance scores are determined, we can now use our linear regression equations in order to predict, with SME input, future scores. Given below in Figure 5 are the results of a small "proof of principle" example consisting of only 20 events with randomly generated scores, each occurring randomly over 200 time steps. These graphs show the cumulative change for each of the four issue stances over time.

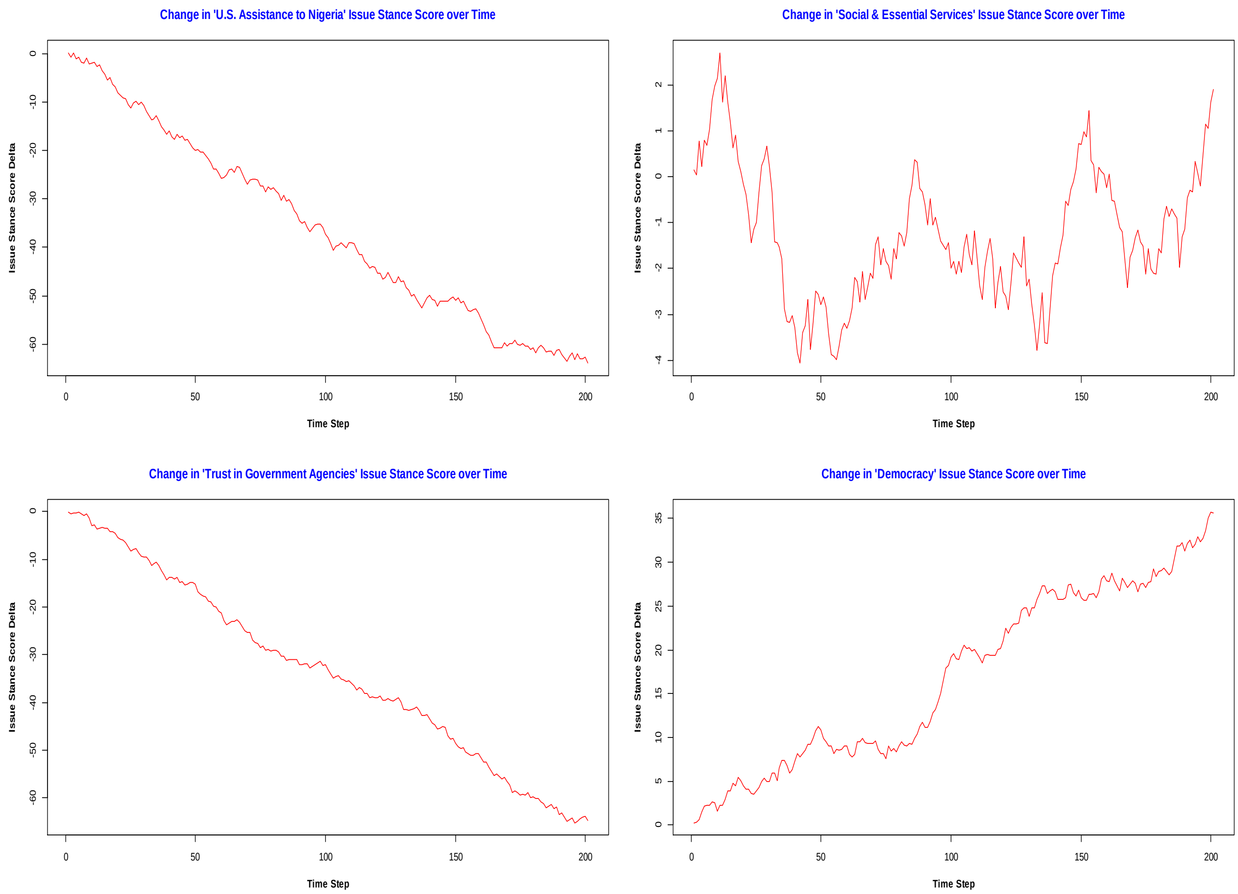

Figure 5. Cumulative issue stance score over time for the 4 key issues.

We can see from the graphs that our randomly generated events have made the population's issue stance concerning "U.S. Assistance to Nigeria" and "Trust in Government Agencies" both decrease over time, while "Social & Essential Services" and "Democracy" see an upward trend. Shown below in Figure 6 is a brief listing of events (including the first and last 25) by time step and the change in each issue stance score.

Time	Event	X2	X4	X5	X10
0	0	0.178	0.151	-0.145	0.272
1	16	-0.859	0.046	-0.482	0.331
2	2	0.131	0.79	-0.4	0.578
3	20	-1.193	0.227	-0.354	1.51
4	19	-0.859	0.793	-0.194	2.178
5	16	-1.896	0.688	-0.531	2.237
6	12	-2.005	1.035	-0.77	2.284
7	11	-0.928	1.696	-0.435	2.631
8	10	-2.203	1.983	-1.326	2.567
9	17	-2.081	2.149	-2.951	1.597
10	19	-1.747	2.715	-2.791	2.265
11	15	-2.661	1.635	-3.632	2.256
12	19	-2.327	2.201	-3.472	2.924
13	20	-3.651	1.638	-3.426	3.856
14	4	-4.224	1.194	-3.502	3.878
15	20	-5.548	0.631	-3.456	4.81
16	13	-5.018	0.906	-4.224	4.486
17	20	-6.342	0.343	-4.178	5.418
18	7	-6.854	0.125	-4.561	5.043
19	5	-8.141	-0.16	-5.498	4.452
20	7	-8.653	-0.378	-5.881	4.077
21	4	-9.226	-0.822	-5.957	4.099
22	9	-9.27	-1.43	-6.456	3.602
23	10	-10.545	-1.143	-7.347	3.538
24	8	-11.291	-0.985	-8.252	3.918
25	11	-10.214	-0.324	-7.917	4.265
176	4	-60.405	-2.004	-59.322	27.751
177	1	-60.328	-2.098	-58.959	29.19
178	3	-61.104	-2.126	-59.968	28.321
179	19	-60.77	-1.56	-59.808	28.989
180	16	-61.807	-1.665	-60.145	29.048
181	2	-60.817	-0.921	-60.063	29.295
182	13	-60.287	-0.646	-60.831	28.971
183	7	-60.799	-0.864	-61.214	28.596
184	8	-61.545	-0.706	-62.119	28.976
185	1	-61.468	-0.8	-61.756	30.415
186	1	-61.391	-0.894	-61.393	31.854
187	15	-62.305	-1.974	-62.234	31.845
188	11	-61.228	-1.313	-61.899	32.192
189	17	-61.106	-1.147	-63.524	31.222
190	18	-62.096	-0.45	-63.047	32.145
191	8	-62.842	-0.292	-63.952	32.525
192	3	-63.618	-0.32	-64.961	31.656
193	11	-62.541	0.341	-64.626	32.003
194	6	-61.831	0.084	-64.246	32.925
195	5	-63.118	-0.201	-65.183	32.334
196	11	-62.041	0.46	-64.848	32.681
197	18	-63.031	1.157	-64.371	33.604
198	1	-62.954	1.063	-64.008	35.043
199	19	-62.62	1.629	-63.848	35.711
200	10	-63.895	1.916	-64.739	35.647

Figure 6. Partial listing of cumulative issue stance changes over time

Turning our attention now to predicting future OAB probabilities using the multinomial logistic regression equations developed in the previous section, and using the same 20 events across 200 time steps as described above, we can look at how the OABs toward each actor change over time as seen in Figure 7.

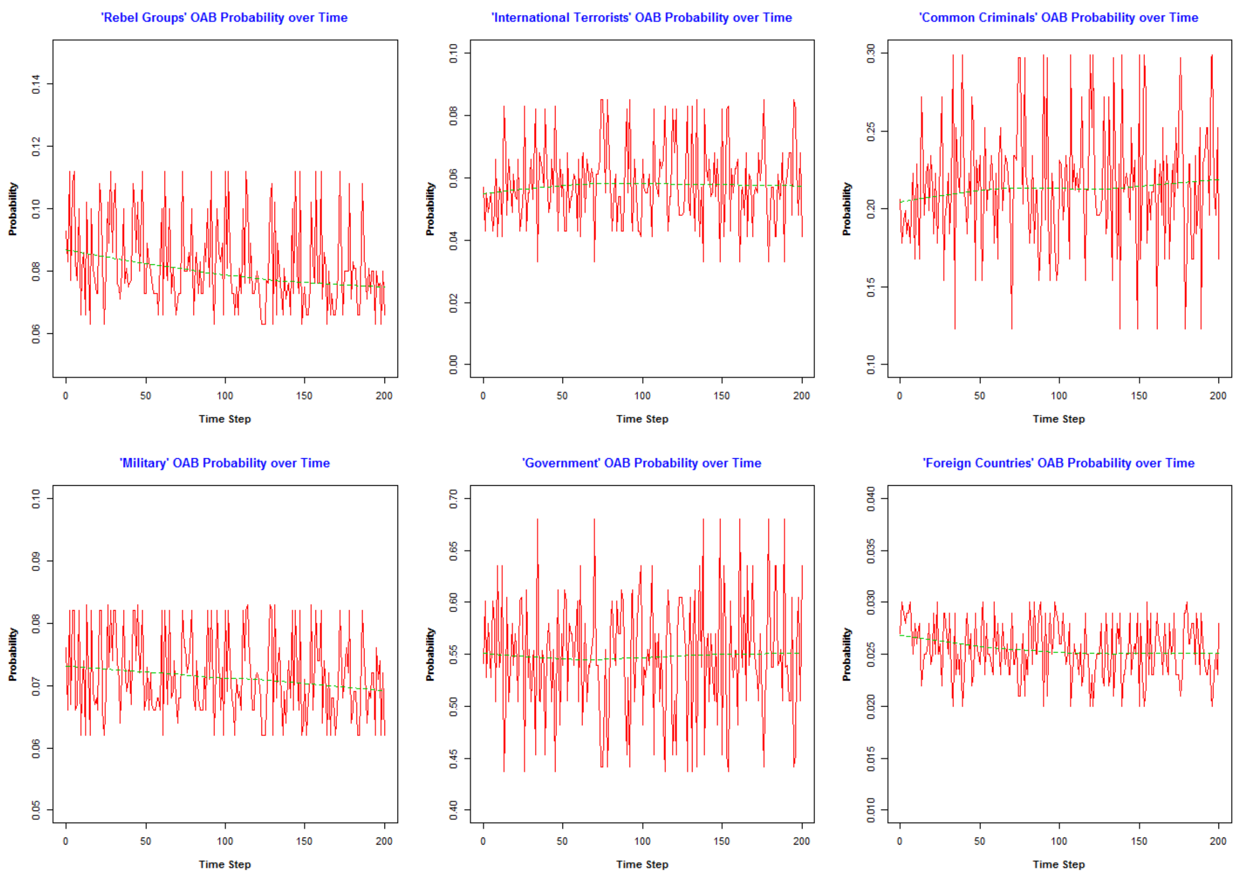

Figure 7. Observed attitude and behavior probabilities over time.

The "Government" OAB has the most variation over time, while the others tended to stay relatively close to their initial value. This is primarily due to the fact that an overwhelming majority of survey respondents had selected "Government Officials" as the primary source of blame for the ongoing violence in their country. Shown below in Figure 8 is a listing of events (including the first and last 25) by time step and the change in each OAB.

Time	Event	Rebel_Groups	International_Terrorists	Common_Criminals	Military	Government	Foreign_Countries
0	0	0.093	0.057	0.206	0.076	0.541	0.027
1	16	0.073	0.061	0.234	0.068	0.539	0.024
2	2	0.108	0.053	0.184	0.082	0.545	0.029
3	20	0.075	0.082	0.299	0.071	0.453	0.02
4	19	0.1	0.066	0.223	0.082	0.504	0.025
5	16	0.073	0.061	0.234	0.068	0.539	0.024
6	12	0.086	0.055	0.203	0.074	0.555	0.027
7	11	0.112	0.055	0.193	0.082	0.528	0.029
8	10	0.066	0.059	0.213	0.068	0.57	0.023
9	17	0.071	0.033	0.123	0.064	0.68	0.029
10	19	0.1	0.066	0.223	0.082	0.504	0.025
11	15	0.073	0.054	0.212	0.066	0.57	0.025
12	19	0.1	0.066	0.223	0.082	0.504	0.025
13	20	0.075	0.082	0.299	0.071	0.453	0.02
14	4	0.081	0.057	0.229	0.069	0.538	0.026
15	20	0.075	0.082	0.299	0.071	0.453	0.02
16	13	0.089	0.043	0.154	0.073	0.612	0.029
17	20	0.075	0.082	0.299	0.071	0.453	0.02
18	7	0.077	0.049	0.199	0.067	0.58	0.028
19	5	0.063	0.048	0.196	0.062	0.605	0.025
20	7	0.077	0.049	0.199	0.067	0.58	0.028
21	4	0.081	0.057	0.229	0.069	0.538	0.026
22	9	0.083	0.043	0.178	0.066	0.601	0.03
23	10	0.066	0.059	0.213	0.068	0.57	0.023
24	8	0.076	0.063	0.211	0.074	0.554	0.023
25	11	0.112	0.055	0.193	0.082	0.528	0.029
176	4	0.081	0.057	0.229	0.069	0.538	0.026
177	1	0.102	0.083	0.272	0.083	0.437	0.022
178	3	0.066	0.041	0.168	0.062	0.635	0.028
179	19	0.1	0.066	0.223	0.082	0.504	0.025
180	16	0.073	0.061	0.234	0.068	0.539	0.024
181	2	0.108	0.053	0.184	0.082	0.545	0.029
182	13	0.089	0.043	0.154	0.073	0.612	0.029
183	7	0.077	0.049	0.199	0.067	0.58	0.028
184	8	0.076	0.063	0.211	0.074	0.554	0.023
185	1	0.102	0.083	0.272	0.083	0.437	0.022
186	1	0.102	0.083	0.272	0.083	0.437	0.022
187	15	0.073	0.054	0.212	0.066	0.57	0.025
188	11	0.112	0.055	0.193	0.082	0.528	0.029
189	17	0.071	0.033	0.123	0.064	0.68	0.029
190	18	0.08	0.085	0.297	0.076	0.442	0.021
191	8	0.076	0.063	0.211	0.074	0.554	0.023
192	3	0.066	0.041	0.168	0.062	0.635	0.028
193	11	0.112	0.055	0.193	0.082	0.528	0.029
194	6	0.112	0.066	0.231	0.082	0.482	0.026
195	5	0.063	0.048	0.196	0.062	0.605	0.025
196	11	0.112	0.055	0.193	0.082	0.528	0.029
197	18	0.08	0.085	0.297	0.076	0.442	0.021
198	1	0.102	0.083	0.272	0.083	0.437	0.022
199	19	0.1	0.066	0.223	0.082	0.504	0.025
200	10	0.066	0.059	0.213	0.068	0.57	0.023

Figure 8. Partial listing of observed attitude and behavior probabilities over time

For both the predicted issue stance scores as well as the OAB probabilities, the event driven values in the form of a look-up table are available in Appendix D.

APPENDIX D. SME INPUT & LOOK-UP TABLE

Event	X1	X3	X6	X7	X8	X9	X11	X12	X13	X14	X15	X16	X17	X18	X19	X20	X21	X22	X23	X24	X25	X26	Safety	Goals	Services	Equality
1	-2	0	-1	0	-2	1	1	0	-2	1	1	0	-2	2	-1	-1	2	1	-2	0	0	-2	1	0	1	-1
2	0	2	2	0	0	0	-2	2	-2	-1	2	1	2	-2	0	2	1	-1	-1	0	0	2	-1	2	-1	0
3	0	-1	1	-2	2	-1	2	2	-1	0	-1	0	0	0	0	0	0	1	0	0	1	0	0	1	-2	0
4	0	-1	1	-2	2	1	0	2	1	1	0	-1	2	-2	1	0	2	1	0	-1	-2	-2	0	2	0	0
5	2	-2	-2	2	-2	-2	-2	2	1	-2	0	1	-2	0	-2	0	0	0	1	2	2	1	0	-2	-2	-2
6	0	2	2	-1	-1	1	2	0	-1	-2	1	0	1	1	-2	-1	2	-2	1	0	0	1	0	-1	-1	-2
7	0	-1	0	2	0	2	-2	0	0	-2	-1	1	2	2	2	0	-2	0	0	-1	2	0	-2	-1	0	2
8	-2	-2	-1	0	-1	-1	0	-2	2	2	0	1	0	-2	0	1	1	0	1	2	2	-2	1	0	2	0
9	2	0	-2	0	0	-1	-1	2	2	0	1	1	0	1	1	1	0	-1	0	1	-1	-2	-1	-2	0	1
10	0	-2	-1	0	1	0	-1	0	0	-2	0	-1	1	-2	0	1	-1	1	-2	1	-1	0	2	0	2	0
11	-1	2	0	0	0	0	0	2	0	2	2	0	0	2	2	0	1	0	0	0	-1	0	0	1	0	0
12	1	1	2	1	-1	0	-2	0	-2	-2	-2	-1	1	-1	0	-1	0	-1	0	1	0	1	0	0	-1	2
13	-2	2	2	-1	1	-2	-2	-2	0	0	2	0	2	-1	2	0	0	1	0	1	-2	-1	0	0	2	-1
14	-2	0	-1	1	0	1	0	0	-2	-2	2	0	-2	-2	-1	0	1	-2	0	2	1	0	1	0	-2	-1
15	2	-2	1	2	0	2	1	-2	2	-1	2	2	1	-2	-2	1	-1	-2	1	-1	0	0	-1	-1	0	-2
16	0	-1	0	0	0	-2	-2	-1	0	0	0	2	-2	-2	2	-1	2	2	0	0	0	-1	0	-2	0	0
17	-2	0	1	1	-1	-2	0	-1	0	-2	-1	-2	-1	1	-1	2	-2	2	1	0	2	0	0	2	0	-1
18	1	-2	2	2	0	0	1	0	1	0	0	0	0	1	0	-2	2	-2	0	2	0	2	0	2	0	-1
19	0	2	-1	-2	-2	1	-2	1	-2	-1	-1	0	1	0	1	-2	2	0	2	2	-1	0	1	1	-1	-2
20	2	-2	1	1	-2	0	-2	-1	0	-2	-2	1	2	-1	0	-2	2	1	-2	2	-1	-2	0	0	0	2

Figure 9. Notional SME input values for each factor by event

Event	X2	X4	X5	X10	Rebels	Terrorists	Criminals	Military	Government	Foreign
1	0.077	-0.094	0.363	1.439	0.102	0.083	0.272	0.083	0.437	0.022
2	0.99	0.744	0.082	0.247	0.108	0.053	0.184	0.082	0.545	0.029
3	-0.776	-0.028	-1.009	-0.869	0.066	0.041	0.168	0.062	0.635	0.028
4	-0.573	-0.444	-0.076	0.022	0.081	0.057	0.229	0.069	0.538	0.026
5	-1.287	-0.285	-0.937	-0.591	0.063	0.048	0.196	0.062	0.605	0.025
6	0.71	-0.257	0.38	0.922	0.112	0.066	0.231	0.082	0.482	0.026
7	-0.512	-0.218	-0.383	-0.375	0.077	0.049	0.199	0.067	0.58	0.028
8	-0.746	0.158	-0.905	0.38	0.076	0.063	0.211	0.074	0.554	0.023
9	-0.044	-0.608	-0.499	-0.497	0.083	0.043	0.178	0.066	0.601	0.03
10	-1.275	0.287	-0.891	-0.064	0.066	0.059	0.213	0.068	0.57	0.023
11	1.077	0.661	0.335	0.347	0.112	0.055	0.193	0.082	0.528	0.029
12	-0.109	0.347	-0.239	0.047	0.086	0.055	0.203	0.074	0.555	0.027
13	0.53	0.275	-0.768	-0.324	0.089	0.043	0.154	0.073	0.612	0.029
14	-0.804	-0.069	-0.057	0.446	0.08	0.068	0.252	0.072	0.505	0.023
15	-0.914	-1.08	-0.841	-0.009	0.073	0.054	0.212	0.066	0.57	0.025
16	-1.037	-0.105	-0.337	0.059	0.073	0.061	0.234	0.068	0.539	0.024
17	0.122	0.166	-1.625	-0.97	0.071	0.033	0.123	0.064	0.68	0.029
18	-0.99	0.697	0.477	0.923	0.08	0.085	0.297	0.076	0.442	0.021
19	0.334	0.566	0.16	0.668	0.1	0.066	0.223	0.082	0.504	0.025
20	-1.324	-0.563	0.046	0.932	0.075	0.082	0.299	0.071	0.453	0.02

Figure 10. Look-up table for issue stance and OAB by event

APPENDIX E. R CODE FOR FACTOR ANALYSIS AND REGRESSION MODELS

There are five distinct pieces of R code that follow, one each for the recoding/imputation of the data, the factor analysis, recoding the response variables, building the models, and a script that will manipulate the data as well as generate plots for the use case implementation.

I. Data Recode and Imputation

```
## Script for recoding and imputing the 2010 Sahel (Nigeria) Survey Data
## This program will output 3 files:
## 1. A recoded data set according to the recode functions listed below
## 2. A recoded and imputed data set using hot decking
## 3. A final data set with only the variables (questions) necessary for factor analysis

## Read in .sav file
library(foreign)

nig10 <- read.spss("C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Nigeria_2010_weights.sav",use.value.labels=TRUE,max.value.labels=Inf, data.frame=TRUE)

# Replace ":" in d17 with "-"
nig10$d17 <- gsub(":","-",nig10$d17)

# Delete BLANK1-BLANK15,"hh_1-7_1-12","reas1-12","length_int",and "sexagewgt"
data <- nig10[,-
c(11,18:101,151,196,199,228,230,235,237,240,242,265,288,333,345,347,358:369,370,375)]

## Recoding scheme based off of number of points in Likert Scale (-2 to 2, "Don't Know = 0")

library(car) # package for recoding

# 1. Two point questions where "Yes" is most positive (Don't Know = 0)
recodeTwoPos <- function(x){
        recode(x,
               '"Yes"= 2;
               "No"= -2;
               "The government serves the interests of all people equally" = 2;
               "The government favors certain groups over others"= -2;
               "Does not affect the amount of crime in society"= 2;
               "Reduces crime in society"= -2;
               "Promotes harsh criminal punishments"= 2;
               "Promotes fair criminal punishments"= -2;
               "Does not affect the amount of corruption in society"= 2;
               "Reduces corruption in society"= -2;
               "Denies women rights"= 2;
               "Denies women\'s rights"= 2;
               "Protects women"= -2;
               "Protects women\'s rights"= -2;
               "Does not treat women as equals to men"= 2;
               "Does not treat women  as equals to men"= 2;
               "Treats women as equals to men"= -2;
               "Treats women as equals to men"= -2;
               "Non-Muslims in Nigeria should be free to worship in their own way"= 2;
               "Non-Muslims in [COUNTRY] should be free to worship in their own way."= 2;
               "Non-Muslims in Nigeria should not be free to worship in their own way"= -2;
               "Non-Muslims in [COUNTRY] should not be able to worship in their own way"= -2;
        "Islam teaches people to deal with non-believers with cooperation and understanding"= 2;
        "Islam teaches people to deal with nonbelievers with cooperation and understanding"= 2;
        "Islam teaches people to deal with non-believers with confrontation and struggle"= -2;
        "Islam teaches people to deal with nonbelievers with confrontation and struggle."= -2;
               "Non-Muslim and Muslim cultures can peacefully exist together" = 2;
```

```
                "Non-Muslim and Muslim cultures can peacefully exist together."= 2;
                "War between Non-Muslim and Muslim cultures is inevitable"= -2;
                "War between Non-Muslim and Muslim cultures is inevitable."= -2;
                "European/American culture is not a threat to traditional Muslim values"= 2;
                "European/American culture is not a threat to traditional Muslim values."= 2;
                "European/American culture is a threat to traditional Muslim values"= -2;
                "European/American culture is a threat to traditional Muslim values."= -2;
                "Muslims who live in France are free to practice Islam"= 2;
                "Muslims who live in France are free to practice Islam."= 2;
                "Muslims who live in France cannot freely practice Islam"= -2;
                "Muslims who live in France cannot freely practice Islam."= -2;
                "Muslims who live in the United States are free to practice Islam"= 2;
                "Muslims who live in the United States of America are free to practice Islam."= 2;
                "Muslims who live in the United States cannot freely practice Islam"= -2;
        "Muslims who live in the United States of America cannot freely practice Islam."= -2;
                "Muslims are treated fairly in the world today"= 2;
                "Muslims are treated fairly in the world today."= 2;
                "Muslims are being oppressed in the world today"= -2;
                "Muslims are being oppressed in the world today."= -2;
"The office of the president should be held by the person most capable regardless of their
regional origin"= 2;
"The office of the president should be alternately held by a notherner and a southerner"= -2;
                "Marabouts sending young boys into the street is a form of exploitation."= 2;
"Marabouts sending young boys into the street is a necessary part of their religious education."=
-2;
                "Don\'t know" = 0;
                "Don\'t Know" = 0;
                "Dont know"= 0;
                "DK"= 0;
                "No answer"= NA;
                "No response"= NA;
                "No repsonse"= NA;
                "No Response"= NA;
                "NR"= NA; ',
                        as.factor.result=FALSE)
}

# 2. Two point questions where "Yes" is most positive (Don't Know = NA)
recodeTwoPos1 <- function(x){
        recode(x,
                '"Yes"= 2;
                "No"= -2;
                "The government serves the interests of all people equally" = 2;
                "The government favors certain groups over others"= -2;
                "Does not affect the amount of crime in society"= 2;
                "Reduces crime in society"= -2;
                "Promotes harsh criminal punishments"= 2;
                "Promotes fair criminal punishments"= -2;
                "Does not affect the amount of corruption in society"= 2;
                "Reduces corruption in society"= -2;
                "Denies women rights"= 2;
                "Denies women\'s rights"= 2;
                "Protects women"= -2;
                "Protects women\'s rights"= -2;
                "Does not treat women as equals to men"= 2;
                "Does not treat women  as equals to men"= 2;
                "Treats women as equals to men"= -2;
                "Treats women as equals to men"= -2;
                "Non-Muslims in Nigeria should be free to worship in their own way"= 2;
                "Non-Muslims in [COUNTRY] should be free to worship in their own way."= 2;
                "Non-Muslims in Nigeria should not be free to worship in their own way"= -2;
                "Non-Muslims in [COUNTRY] should not be able to worship in their own way"= -2;
        "Islam teaches people to deal with non-believers with cooperation and understanding"= 2;
        "Islam teaches people to deal with nonbelievers with cooperation and understanding"= 2;
        "Islam teaches people to deal with non-believers with confrontation and struggle"= -2;
        "Islam teaches people to deal with nonbelievers with confrontation and struggle."= -2;
                "Non-Muslim and Muslim cultures can peacefully exist together" = 2;
                "Non-Muslim and Muslim cultures can peacefully exist together."= 2;
                "War between Non-Muslim and Muslim cultures is inevitable"= -2;
```

```
                "War between Non-Muslim and Muslim cultures is inevitable."= -2;
                "European/American culture is not a threat to traditional Muslim values"= 2;
                "European/American culture is not a threat to traditional Muslim values."= 2;
                "European/American culture is a threat to traditional Muslim values"= -2;
                "European/American culture is a threat to traditional Muslim values."= -2;
                "Muslims who live in France are free to practice Islam"= 2;
                "Muslims who live in France are free to practice Islam."= 2;
                "Muslims who live in France cannot freely practice Islam"= -2;
                "Muslims who live in France cannot freely practice Islam."= -2;
                "Muslims who live in the United States are free to practice Islam"= 2;
                "Muslims who live in the United States of America are free to practice Islam."= 2;
                "Muslims who live in the United States cannot freely practice Islam"= -2;
        "Muslims who live in the United States of America cannot freely practice Islam."= -2;
                "Muslims are treated fairly in the world today"= 2;
                "Muslims are treated fairly in the world today."= 2;
                "Muslims are being oppressed in the world today"= -2;
                "Muslims are being oppressed in the world today."= -2;
"The office of the president should be held by the person most capable regardless of their
regional origin"= 2;
"The office of the president should be alternately held by a notherner and a southerner"= -2;
                "Marabouts sending young boys into the street is a form of exploitation."= 2;
"Marabouts sending young boys into the street is a necessary part of their religious education."=
-2;
                "Don\'t know" = NA;
                "Don\'t Know" = NA;
                "Dont know"= NA;
                "DK"= NA;
                "No answer"= NA;
                "No response"= NA;
                "No repsonse"= NA;
                "No Response"= NA;
                "NR"= NA; ',
                        as.factor.result=FALSE)
}

# 3. Two point questions where "Yes" is most negative ("No" and "Oppose" is positive, Don't Know
= 0)
recodeTwoNeg <- function(x){
        recode(x,
                '"Yes"= -2;
                "No"= 2;
                "Oppose"= 2;
                "Support"= -2;
                "Justified"= -2;
                "Not Justified"= 2;
                "Don\'t know" = 0;
                "Don\'t Know" = 0;
                "Dont know"= 0;
                "DK"= 0;
                "No answer"= NA;
                "No response"= NA;
                "No Response"= NA;
                "No repsonse"= NA;
                "NR"= NA; ',
                        as.factor.result=FALSE)
}

# 4. Two point questions where "Yes" is most negative ("No" and "Oppose" is positive, Don't Know
= NA)
recodeTwoNeg1<- function(x){
        recode(x,
                '"Yes"= -2;
                "No"= 2;
                "Oppose"= 2;
                "Support"= -2;
                "Justified"= -2;
                "Not Justified"= 2;
                "Don\'t know" = NA;
                "Don\'t Know" = NA;
```

```
                "Dont know"= NA;
                "DK"= NA;
                "No answer"= NA;
                "No response"= NA;
                "No Response"= NA;
                "No repsonse"= NA;
                "NR"= NA; ',
                        as.factor.result=FALSE)
}

# 5. Three point questions where "Most" is preferred (positive)
recodeThreePos <- function(x){
        recode(x,
                '"Improved"= 2;
                "Stayed the same"=0;
                "Gotten worse"=-2;
                "Worsened"=-2;
                "Government and religion should be kept separate" = 2;
                "Government and religion should be kept separate."= 2;
"Our country should remain a secular democracy, but religion should play a greater role in govt"=
0;
"Our country should remain a secular democracy, but religion should play a greater role in
government."=0;
                "Our country should be governed by religious leaders"= -2;
                "Our country should be governed by religious leaders." = -2;
                "Our country should be governed by civil law" =2;
                "Our country should be governed by civil law." =2;
                "Our country should be gvoerned by a combination of civil and religious law"= 0;
                "Our country should be governed by a combination of civil and religious law."=0;
                "Religious laws should govern all spheres of life"= -2;
                "Jihad is an inward personal and moral struggle"= 2;
                "Jihad is both"= 0;
                "Jihad is taking up arms against enemies of Islam"= -2;
                "The U.S is engaged to fight terrorism"= 2;
                "The U.S. is engaged to fight terrorism"= 2;
                "Both"= 0;
                "None"= 0;
                "The U.S. is engaged to fight Islam"= -2;
                "Dont know"= NA;
                "No response"= NA;
                "No repsonse"= NA;
                "No answer"= NA;
                "Dont know"= NA;
                "Don\'t know"= NA;
                "Don\'t Know"= NA;
                "No Response"= NA;
                "No response"= NA;
                "No Response"= NA;
                "No answer"= NA;',
                        as.factor.result=FALSE)
}

# 6. Three point questions where "Most" is least preferred (negative)
recodeThreeNeg <- function(x){
        recode(x,
                '"Positive influence"= -2;
                "Neutral influence"= 0;
                "Negative influence"= 2;
                "No influence"= 0;
                "Don\'t know"= NA;
                "Dont Know"= NA;
                "No Response"= NA;
                "No repsonse"= NA;
                "No response"= NA; ',
                        as.factor.result=FALSE)
}

# 7. Four point questions where "Most" is preferred (positive)
recodeFourPos <- function(x){
```

```
recode(x,
        '"Never"=-2;
        "Several times a year"= -1;
        "Several times a month" = 1;
        "Several times a week"= 2;
        "Very safe"= 2;
        "Fairly safe"= 1;
        "Not very safe"= -1;
        "Not safe at all"= -2;
        "Very satisified"= 2;
        "Very satisfied"= 2;
        "Somewhat satisfied"= 1;
        "Not very satisfied"= -1;
        "Somewhat frustrated"= -1;
        "Not at all satisfied"= -2;
        "Very frustrated"= -2;
        "Nigeria is not a democracy"= -2;
        "Very favorable"= 2;
        "Somewhat favorable"= 1;
        "Somewhat unfavorable"= -1;
        "Very unfavorable"= -2;
        "Very similar"= 2;
        "Somewhat similar"= 1;
        "Only a little similar"= -1;
        "Not similar at all"= -2;
        "A lot"= 2;
        "A Lot"= 2;
        "A fair amount"= 1;
        "A Fair amount"= 1;
        "Fair amount"= 1;
        "A little"= -1;
        "Not at all"= -2;
        "No trust at all"= -2;
        "A lot of confidence"= 2;
        "A fair amount of confidence"= 1;
        "Only little confidence"= -1;
        "Only a little confidence"= -1;
        "No confidence at all"= -2;
        "Very stable"= 2;
        "Somewhat stable"= 1;
        "Somewhat fragile"= -1;
        "Very fragile"= -2;
        "Strongly agree"= 2;
        "Somewhat agree"= 1;
        "Somewhat disagree"= -1;
        "Strongly disagree"= -2;
        "Strongly approve"= 2;
        "Somewhat approve"= 1;
        "Somewhat disapprove"= -1;
        "Strongly disapprove"= -2;
        "Very good"= 2;
        "Somewhat good"= 1;
        "Somewhat poor"= -1;
        "Very poor"= -2;
        "Very good"= 2;
        "Good"= 1;
        "Fair"= -1;
        "Poor"= -2;
        "Often"= 2;
        "Sometimes"= 1;
        "Rarely"= -1;
        "Never"= -2;
        "Very easy"= 2;
        "Somewhat easy"= 1;
        "Somewhat hard"= -1;
        "Very hard"= -2;
        "Very often"= 2;
        "Fairly often"= 1;
        "Not very often"= -1;
```

```
                "Not at all"= -2;
                "Very important"= 2;
                "Fairly important"= 1;
                "Not very important"= -1;
                "Not at all important"= -2;
                "Very responsive"= 2;
                "Somewhat responsive"= 1;
                "Not very responsive"= -1;
                "Not at all responsive"= -2;
                "[COUNTRY]is not a democracy"= NA;
                "NA"= NA;
                "DK"= 0;
                "NR"= NA;
                "No answer"= NA;
                "Don\'t know"= 0;
                "Don\'t Know"= 0;
                "Dont know"= 0;
                "No response"= NA;
                "No response"= NA;
                "No repsonse"= NA;
                "No repsonse"= NA;
                "No Response"= NA; ',
                        as.factor.result=FALSE)
}

# 8. Four point questions where "Most" is least preferred (negative)
recodeFourNeg <- function(x){
        recode(x,
                '"Always justified"= -2;
                "Sometimes justified"= -1;
                "Rarely justified"= 1;
                "Never justified"= 2;
                "Strongly agree"= -2;
                "Somewhat agree"= -1;
                "Somewhat disagree"= 1;
                "Strongly disagree"= 2;
                "Often"= -2;
                "Sometimes"= -1;
                "Rarely"= 1;
                "Never"= 2;
                "DK"= 0;
                "NR"= NA;
                "No answer"= NA;
                "Don\'t know"= 0;
                "Don\'t Know"= 0;
                "Dont know"= 0;
                "No response"= NA;
                "No repsonse"= NA;
                "No Response"= NA; ',
                        as.factor.result=FALSE)
}

# 9. Five point questions where "Most" is preferred (positive)
recodeFivePos <- function(x){
        recode(x,
                '"Always"= 2;
                "Most of every day"= 1;
                "Only a few hours a day"= 0;
                "Only a few hours a week"= -1;
                "Never"= -2;
                "Upper- Plenty of disposable money"= 2;
                "Upper middle- Able to purchase most essential goods"= 1;
                "Lower middle- Able to meet basic needs with some non-essential goods"= -1;
                "Poor- Able to meet basic needs"= -1;
                "Very poor- Unable to meet basic needs without charity"= -2;
                "Plenty of disposable money"= 2;
                "Able to purchase most non-essential goods"= 1;
                "Able to meet basic needs with some non-essential goods"= 0;
                "Able to meet basic needs" = -1;
```

```
                "Unable to meet basic needs without charity"= -2;
                "DK"= NA;
                "NR"= NA;
                "No answer"= NA;
                "Don\'t know"= NA;
                "Don\'t Know"= NA;
                "Dont know"= NA;
                "No response"= NA;
                "No repsonse"= NA;
                "No Response"= NA; ',
                        as.factor.result=FALSE)
}

# 10. Five point questions where "Most" is the least preferred (negative)
recodeFiveNeg <- function(x){
        recode(x,
                '"Increased a lot"= -2;
                "Increased a little"= -1;
                "Stayed the same"= 0;
                "Decreased a little"= 1;
                "Decrease a lot"= 2;
                "Increased dramatically"= -2;
                "Increased slightly" = -1;
                "Stayed the same"= 0;
                "Decreased slightly"= 1;
                "DK"= NA;
                "NR"= NA;
                "No answer"= NA;
                "Don\'t know"= NA;
                "Don\'t Know"= NA;
                "Dont know"= NA;
                "No response"= NA;
                "No repsonse"= NA;
                "No Response"= NA; ',
                        as.factor.result=FALSE)
}

# These are recoded for imputation purposes as the Match.var variable. Others may be included.
recodeDem <- function(x){
        recode(x,
                '"Christianity"= 1;
                "Christianity (Catholic, Protestant, Evangelical, etc)"= 1;
                "Islam"= -1;
                "Traditional"= 0;
                "No religion"= 0;
                "Others"= 0;
                "Other"= 0;
                "Judaism"= 0;
                "Animism"= 0;
                "Missing" = 0;
                "No Response"= 0;
                "No response"= 0;
                "Don\'t know"= 0;
                "Male"= 1;
                "Female"= -1;
                "Rural"= 1;
                "Urban"= -1; ',
                        as.factor.result=FALSE)
}

# Recode Question 47 for model building purposes
recodeQ47 <- function(x){
        recode(x,
                '"Rebel groups"= 0;
                "International terrorists"= 1;
                "Common criminals"= 2;
                "The military"= 3;
                "Government officials"= 4;
                "Foreign country"= 5;
```

```
                "Other"= NA;
                "Don\'t know"= NA;
                "No Response"= NA; ',
                        as.factor.result=FALSE)
}

# Link each question to specific recode functions and recode

data$urbanrural <- as.numeric(recodeDem(data$urbanrural))
data$q5 <- as.numeric(recodeFourPos(data$q5))
data$q6 <- as.numeric(recodeFourPos(data$q6))
data$q7 <- as.numeric(recodeFourPos(data$q7))
data$q8edu <- as.numeric(recodeFourPos(data$q8edu))
data$q8hea <- as.numeric(recodeFourPos(data$q8hea))
data$q8wat <- as.numeric(recodeFourPos(data$q8wat))
data$q8roa <- as.numeric(recodeFourPos(data$q8roa))
data$q8ele <- as.numeric(recodeFourPos(data$q8ele))
data$q9edu <- as.numeric(recodeThreePos(data$q9edu))
data$q9hea <- as.numeric(recodeThreePos(data$q9hea))
data$q9wat <- as.numeric(recodeThreePos(data$q9wat))
data$q9roa <- as.numeric(recodeThreePos(data$q9roa))
data$q9ele <- as.numeric(recodeThreePos(data$q9ele))
data$q10 <- as.numeric(recodeTwoPos(data$q10))
data$q12uk <- as.numeric(recodeFourPos(data$q12uk))
data$q12fr <- as.numeric(recodeFourPos(data$q12fr))
data$q12ni <- as.numeric(recodeFourPos(data$q12ni))
data$q12ir <- as.numeric(recodeFourPos(data$q12ir))
data$q12ch <- as.numeric(recodeFourPos(data$q12ch))
data$q14usa <- as.numeric(recodeFourPos(data$q14usa))
data$q16so <- as.numeric(recodeFourPos(data$q16so))
data$q16li <- as.numeric(recodeFourPos(data$q16li))
data$q16sa <- as.numeric(recodeFourPos(data$q16sa))
data$q17sa <- as.numeric(recodeFourPos(data$q17sa))
data$q17fr <- as.numeric(recodeFourPos(data$q17fr))
data$q17ch <- as.numeric(recodeFourPos(data$q17ch))
data$q17ir <- as.numeric(recodeFourPos(data$q17ir))
data$q17us <- as.numeric(recodeFourPos(data$q17us))
data$q21a <- as.numeric(recodeFourPos(data$q21a))
data$q21b <- as.numeric(recodeFourPos(data$q21b))
data$q21c <- as.numeric(recodeFourPos(data$q21c))
data$q21d <- as.numeric(recodeFourPos(data$q21d))
data$q21e <- as.numeric(recodeFourPos(data$q21e))
data$q21f <- as.numeric(recodeFourPos(data$q21f))
data$q21g <- as.numeric(recodeFourPos(data$q21g))
data$q21h <- as.numeric(recodeFourPos(data$q21h))
data$q22a <- as.numeric(recodeFourPos(data$q22a))
data$q22b <- as.numeric(recodeFourPos(data$q22b))
data$q22c <- as.numeric(recodeFourPos(data$q22c))
data$q22d <- as.numeric(recodeFourPos(data$q22d))
data$q22e <- as.numeric(recodeFourPos(data$q22e))
data$q22f <- as.numeric(recodeFourPos(data$q22f))
data$q22g <- as.numeric(recodeFourPos(data$q22g))
data$q22h <- as.numeric(recodeFourPos(data$q22h))
data$q23a <- as.numeric(recodeFourPos(data$q23a))
data$q23b <- as.numeric(recodeFourPos(data$q23b))
data$q23c <- as.numeric(recodeFourPos(data$q23c))
data$q23d <- as.numeric(recodeFourPos(data$q23d))
data$q23e <- as.numeric(recodeFourPos(data$q23e))
data$q23f <- as.numeric(recodeFourPos(data$q23f))
data$q25a <- as.numeric(recodeFourNeg(data$q25a))
data$q25b <- as.numeric(recodeFourNeg(data$q25b))
data$q25c <- as.numeric(recodeFourNeg(data$q25c))
data$d5a <- as.numeric(recodeDem(data$d5a))
data$q26a <- as.numeric(recodeFourPos(data$q26a))
data$q26b <- as.numeric(recodeFourPos(data$q26b))
data$q26c <- as.numeric(recodeFourPos(data$q26c))
data$q26d <- as.numeric(recodeFourPos(data$q26d))
data$q26e <- as.numeric(recodeFourPos(data$q26e))
data$q27a <- as.numeric(recodeTwoPos1(data$q27a))
```

```
data$q27b <- as.numeric(recodeTwoPos1(data$q27b))
data$q27c <- as.numeric(recodeTwoPos1(data$q27c))
data$q27d <- as.numeric(recodeTwoPos1(data$q27d))
data$q28 <- as.numeric(recodeThreePos(data$q28))
data$q29a <- as.numeric(recodeTwoPos(data$q29a))
data$q29b <- as.numeric(recodeTwoPos(data$q29b))
data$q29c <- as.numeric(recodeTwoPos(data$q29c))
data$q29d <- as.numeric(recodeTwoPos(data$q29d))
data$q30 <- as.numeric(recodeThreePos(data$q30))
data$q31a <- as.numeric(recodeThreePos(data$q31a))
data$q31b <- as.numeric(recodeThreePos(data$q31b))
data$q32a <- as.numeric(recodeTwoPos(data$q32a))
data$q32b <- as.numeric(recodeTwoPos(data$q32b))
data$q32c <- as.numeric(recodeTwoPos(data$q32c))
data$q32d <- as.numeric(recodeTwoPos(data$q32d))
data$q32e <- as.numeric(recodeTwoPos(data$q32e))
data$q33 <- as.numeric(recodeTwoNeg(data$q33))
data$q34a <- as.numeric(recodeTwoPos(data$q34a))
data$q34b <- as.numeric(recodeTwoPos(data$q34b))
data$q36a <- as.numeric(recodeFourNeg(data$q36a))
data$q36b <- as.numeric(recodeFourNeg(data$q36b))
data$q37a <- as.numeric(recodeTwoPos(data$q37a))
data$q37b <- as.numeric(recodeTwoPos(data$q37b))
data$q37c <- as.numeric(recodeTwoPos(data$q37c))
data$q37d <- as.numeric(recodeTwoPos(data$q37d))
data$q37e <- as.numeric(recodeTwoPos(data$q37e))
data$q40 <- as.numeric(recodeFourPos(data$q40))
data$q41a <- as.numeric(recodeTwoPos1(data$q41a))# Don't know=0 here because it is not an opinion
data$q42 <- as.numeric(recodeFourPos(data$q42))
data$q44 <- as.numeric(recodeFourPos(data$q44))
data$q44na <- as.numeric(recodeTwoPos(data$q44na))
data$q45 <- as.numeric(recodeFourPos(data$q45))
data$q47 <- as.numeric(recodeQ47(data$q47))
data$q48a <- as.numeric(recodeFourPos(data$q48a))
data$q48b <- as.numeric(recodeFourPos(data$q48b))
data$q48c <- as.numeric(recodeFourPos(data$q48c))
data$q48d <- as.numeric(recodeFourPos(data$q48d))
data$q48e <- as.numeric(recodeFourPos(data$q48e))
data$q48f <- as.numeric(recodeFourPos(data$q48f))
data$q49pr <- as.numeric(recodeFourPos(data$q49pr))
data$q49pm <- as.numeric(recodeFourPos(data$q49pm))
data$q49na <- as.numeric(recodeFourPos(data$q49na))
data$q49pp <- as.numeric(recodeFourPos(data$q49pp))
data$q49af <- as.numeric(recodeFourPos(data$q49af))
data$q49cj <- as.numeric(recodeFourPos(data$q49cj))
data$q49rl <- as.numeric(recodeFourPos(data$q49rl))
data$q49lp <- as.numeric(recodeFourPos(data$q49lp))
data$q49lg <- as.numeric(recodeFourPos(data$q49lg))
data$q50 <- as.numeric(recodeFourPos(data$q50))
data$q52 <- as.numeric(recodeFourPos(data$q52))
data$q56b <- as.numeric(recodeThreePos(data$q56b))
data$q57 <- as.numeric(recodeTwoNeg(data$q57))
data$q58a <- as.numeric(recodeTwoNeg(data$q58a))
data$q58b <- as.numeric(recodeTwoNeg(data$q58b))
data$q58c <- as.numeric(recodeTwoNeg(data$q58c))
data$q59a <- as.numeric(recodeFourPos(data$q59a))
data$q59b <- as.numeric(recodeFourNeg(data$q59b))
data$q59c <- as.numeric(recodeFourNeg(data$q59c))
data$q59d <- as.numeric(recodeFourPos(data$q59d))
data$q60 <- as.numeric(recodeThreePos(data$q60))
data$q62a <- as.numeric(recodeFourPos(data$q62a)) #It’s opinions; can't determine a pos or neg
data$q62b <- as.numeric(recodeFourPos(data$q62b))
data$q62c <- as.numeric(recodeFourPos(data$q62c))
data$q62d <- as.numeric(recodeFourPos(data$q62d))
data$d0 <- as.numeric(recodeDem(data$d0))
data$d13 <- as.numeric(recodeFourPos(data$d13))  # conditional question
data$d15 <- as.numeric(recodeThreePos(data$d15))
data$d16 <- as.numeric(recodeFiveNeg(data$d16))
data$d17 <- as.numeric(recodeFivePos(data$d17))
```

```
data$d21 <- as.numeric(recodeFivePos(data$d21))
data$d22 <- as.numeric(recodeTwoPos1(data$d22)) # Don't Know = NA. No cell phone?
data$d23 <- as.numeric(recodeFourPos(data$d23))
data$d24a <- as.numeric(recodeTwoPos1(data$d24a)) # Don't Know = NA
data$d24b <- as.numeric(recodeTwoPos1(data$d24b))
data$d24c <- as.numeric(recodeTwoPos1(data$d24c))
data$d24d <- as.numeric(recodeTwoPos1(data$d24d))
data$d24e <- as.numeric(recodeTwoNeg1(data$d24e))
data$d26a <- as.numeric(recodeFourPos(data$d26a))
data$d26b <- as.numeric(recodeFourPos(data$d26b))
data$d30 <- as.numeric(recodeFourPos(data$d30))

write.table(data,"C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Recode_10.csv",sep=",",col.names=TRUE,row.names=FALSE,quote=TRUE,na="NA")


## Imputation Using Hotdeck Method

library(StatMatch)

imputeHD <- function(Question,Dframe,Donor.Class,Match.vars){

Data.rec <- Dframe[is.na(Dframe[,Question])==TRUE,]
Data.rec <- subset(Data.rec,select=-get(Question))

Data.don <- Dframe[is.na(Dframe[,Question])==FALSE,]

imp.RAND <- RANDwNND.hotdeck(data.rec=Data.rec,data.don=Data.don,match.vars=Match.vars,
               don.class=Donor.Class,dist.fun="Manhattan")

Data.rec.imp <-
create.fused(data.rec=Data.rec,data.don=Data.don,mtc.ids=imp.RAND$mtc.ids,z.vars=Question)

final <- rbind(Data.rec.imp,Data.don)
return(final)
}

HD.loop <- function (Dframe, Donor.Class, Match.vars, Question) {
       empty <- "False"
       while (empty == "False"){
               final <- imputeHD (Question[1], Dframe, Donor.Class, Match.vars)
               Question <- Question[-1]  # remove that question FIFO
               Dframe <- final  # update Dframe with new data
               if (length(Question) < 1){
                       empty <- "True"
               }
       }
       final
}

Match.vars <- c("d5a","d0","urbanrural")
data$state <- as.factor(data$state) # state must be a factor
Donor.Class <- c("state") #state is the donor class
Dframe <- data
Question <- c("q5","q6","q7","q8edu", "q8hea", "q8wat", "q8roa", "q8ele", "q9edu", "q9hea",
"q9wat","q9roa","q9ele", "q10", "q12uk", "q12fr", "q12ni", "q12ir", "q12ch", "q14usa",
"q16so", "q16li", "q16sa", "q17sa", "q17fr", "q17ch", "q17ir", "q17us", "q21a", "q21b", "q21c",
"q21d", "q21e", "q21f", "q21g", "q21h", "q22a","q22b", "q22c", "q22d", "q22e", "q22f", "q22g",
"q22h","q23a", "q23b", "q23c", "q23d", "q23e", "q23f", "q25a", "q25b", "q25c", "q26a","q26b",
"q26c","q26d", "q26e", "q27a", "q27b", "q27c", "q27d", "q28", "q29a", "q29b", "q29c", "q29d",
"q30","q31a", "q31b", "q32a", "q32b", "q32c", "q32d", "q32e", "q33", "q34a", "q34b", "q36a",
"q36b","q37a", "q37b", "q37c", "q37d", "q40", "q41a", "q42", "q44", "q44na", "q45", "q47",
"q48a", "q48b", "q48c","q48d", "q48e", "q48f", "q49pr", "q49pm", "q49na", "q49pp", "q49af",
"q49cj","q49rl", "q49lp", "q49lg","q50", "q52", "q56b", "q57", "q58a", "q58b", "q58c", "q59a",
"q59b", "q59c", "q59d", "q60", "q62a","q62b", "q62c", "q62d", "d13", "d15", "d16", "d17", "d21",
"d22", "d23", "d24a", "d24b", "d24c", "d24d","d24e", "d26a", "d26b", "d30")

rec.imp.data <- HD.loop(Dframe,Donor.Class,Match.vars,Question)
```

```
write.table(rec.imp.data,"C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Rec_Imp_10.csv",sep=",",col.names=TRUE,row.names=FALSE,quote=TRUE,na="NA")

# Delete all variables except those we want to create factors with (taken from the Questions)
final.data <- rec.imp.data[,c("q5","q6","q7","q8edu", "q8hea", "q8wat", "q8roa", "q8ele",
"q9edu", "q9hea","q9wat","q9roa","q9ele", "q10", "q12uk", "q12fr", "q12ni", "q12ir", "q12ch",
"q14usa","q16so", "q16li", "q16sa", "q17sa", "q17fr", "q17ch", "q17ir", "q17us", "q21a", "q21b",
"q21c","q21d", "q21e", "q21f", "q21g", "q21h", "q22a","q22b", "q22c", "q22d", "q22e", "q22f",
"q22g", "q22h","q23a", "q23b", "q23c", "q23d", "q23e", "q23f", "q25a", "q25b", "q25c",
"q26a","q26b", "q26c","q26d", "q26e", "q27a", "q27b", "q27c", "q27d", "q28", "q29a", "q29b",
"q29c", "q29d", "q30","q31a", "q31b", "q32a", "q32b", "q32c", "q32d", "q32e", "q33", "q34a",
"q34b", "q36a", "q36b","q37a", "q37b", "q37c", "q37d", "q40", "q41a", "q42", "q44", "q44na",
"q45", "q48a", "q48b", "q48c","q48d", "q48e", "q48f", "q49pr", "q49pm", "q49na", "q49pp",
"q49af", "q49cj","q49rl", "q49lp", "q49lg","q50", "q52", "q56b", "q57", "q58a", "q58b", "q58c",
"q59a", "q59b", "q59c", "q59d", "q60", "q62a","q62b", "q62c", "q62d", "d13", "d15", "d16", "d17",
"d21", "d22", "d23", "d24a", "d24b", "d24c", "d24d","d24e", "d26a", "d26b", "d30")]

# Check to see if there are any missing values remaining
for (i in 1:ncol(final.data)) {
        check <- sum(is.na(final.data[,i]))
        # show(check)
}
sum(check)

write.table(final.data,"C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Final_10.csv",sep=",",col.names=TRUE,row.names=FALSE,quote=TRUE,na="NA")
```

II. Factor Analysis

```
## Script for conducting Factor Analysis on the 2010 Sahel (Nigeria) Survey Data
# Function finds optimal number of factors, forms a matrix of the factor loadings as the output.
# Prints out the optimal number of factors used based off of eigenvalues.
# Prints out the factor matrix with loadings > 0.4 or < -0.4.
# Prints out the variable names by factor as well as the factor names.
# Prints the % of variance the factor will explain via eigenvalues.
# Modifies the loading matrix by deleting factors that are n/a.
# Calculates the matrix of factor scores.
# Scales the factor score matrix appropriately to values between -2 and 2.

final.data <- read.csv("C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel Survey/Final_10.csv")

factorNames <- c("1. Sharia Law", "2. U.S. Assist to Nigeria", "3. China Assist to Nigeria", "4.
Social & Essential Services","5. Trust in Gov Agencies", "6. External Security", "7. General
Trust", "8. Non-West Countries", "9. Local and National Freedom","10. Democracy", "11. Others
Values", "12. Daily Life Acceptance", "13. Use of Violence","14. Terrorism Enablers", "15. Family
and Friends", "16. Civic Duty", "17. Attacks on U.S.","18. Discussion of U.S.", "19.
Electricity", "20. Western Countries", "21. Trust in Policy Makers","22. Religious Freedom in the
West", "23. Religious Intolerance", "24. Civility", "25. Policy and Law","26. Roads", "27. None",
"28. None", "29. None")

initial.factor.analysis <- function(data,num){

## Find the optimal number of factors for a field of data
        ev <- eigen(cor(data))
        if(num!=0) {
                num <- num
        }
        else {
                num <- length(ev$values[ev$values > 1])
        }

## Conduct factor analysis
        fact <- factanal(data,factors=num,rotation="varimax")

## Convert the factor loadings to a matrix and name the factors
        fa.mat <- numeric(0)
        for(i in 1:num){
```

```
                fake.fac.load <- fact$loadings[,i]
                fake.fac.load[fact$loadings[,i] < 0.4 & (fact$loadings[,i] > -0.4)] <- 0
                fa.mat <- cbind(fa.mat, fake.fac.load)
        }
        colnames(fa.mat) <- c()
        rownames(fa.mat) <- c()
        rownames(fa.mat) <- c(colnames(data))
        colnames(fa.mat) <- colnames(fa.mat, do.NULL= FALSE, prefix = "Factor.")
        fa.mat # matrix with loadings > 0.4 or < -0.4

## Calculate the variance of each variable

        i.j.MatrixLoc <- which(fa.mat!=0, arr.ind=TRUE)
        z <- tapply (i.j.MatrixLoc[,1], i.j.MatrixLoc[,2],
                        function (x) sum (ev$values[x]))/length(ev$values)
        z <- as.matrix(z)
        dim(z) <- length(z)
        rownames(z) <- rownames(z, do.NULL= FALSE, prefix = "Factor.")

## Print the Output

cat("The number of factors (based off of eigen values or given) is: ", num, "\n",
sep="",file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel Survey/Data10FactorOutput.txt",
append=FALSE)
cat("\n","The number of relevent factors is: ",length(z),"\n", sep="",
file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel Survey/Data10FactorOutput.txt", append=TRUE)
cat("\n","The variables per factor are: ", "\n","=================================",
sep="",file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel Survey/Data10FactorOutput.txt",
append=TRUE)

        x <- numeric(0)
        for(i in 1:ncol(fa.mat)){
                f <- rownames(fa.mat)[which(fa.mat[,i]!=0)]
                x <- fa.mat[which(fa.mat[,i]!=0),i]
                x <- as.matrix(x)
                rownames(f) <- c(colnames(fa.mat[,i]))
                colnames(x) <- c(colnames(fa.mat[,i]))

cat("\n","Factor",i,"= ", sep=" ",
file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel Survey/Data10FactorOutput.txt", append=TRUE)
cat(round(x,4), sep=", ", file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Data10FactorOutput.txt", append=TRUE)
cat("\n","Factor",i,"= ", sep=" ",file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Data10FactorOutput.txt", append=TRUE)
cat(f, sep=", ", file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Data10FactorOutput.txt", append=TRUE)
cat("\n","--------------------------------------------------------------", "\n", sep="",
file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel Survey/Data10FactorOutput.txt", append=TRUE)
}
cat("\n","--------------------------------------------------------------","\n","\n","The
variance impact of each factor is in % : ", "\n",
"===============================================","\n",
sep="",file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel Survey/Data10FactorOutput.txt",
append=TRUE)

write.table(round(z,4)*100,"C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Data10FactorOutput.txt", append=TRUE,sep="= ", col.names=FALSE, row.names=TRUE,
quote=FALSE, na="NA")
}

initial.factor.analysis(final.data,29)

factor.analysis <- function(data,num,name){

        fact <- factanal(data,factors=num,rotation="varimax")

## Convert the factor loadings to a matrix and name the factors
        fa.mat <- numeric(0)
        for(i in 1:num){
```

```
                fake.fac.load <- fact$loadings[,i]
                fake.fac.load[fact$loadings[,i] < 0.4 & (fact$loadings[,i] > -0.4)] <- 0
                fa.mat <- cbind(fa.mat, fake.fac.load) # builds a matrix of factors
        }
        colnames(fa.mat) <- c()
        rownames(fa.mat) <- c()
        rownames(fa.mat) <- c(colnames(data))
        colnames(fa.mat) <- colnames(fa.mat, do.NULL= FALSE, prefix = "Factor.")
        fa.mat # matrix with loadings > 0.4 or < -0.4

        if (is.na(name)==FALSE){
                colnames(fa.mat)<- c(name)
                return(fa.mat)
        }
        else{
        return(fa.mat)
        }
}

Nig.factors <- factor.analysis(final.data,29,factorNames)

## Modify factors & Create Matrix of Factor Scores

Nig.factors <- Nig.factors[,-c(27,28,29)] # delete factors 27, 28, 29
Nig.factors[24,8] <- 0 # delete q17sa in factor 8
Nig.factors[27,8] <- 0 # delete q17ir in factor 8

final.data <- as.matrix(final.data)

factor.scores <- data.frame(final.data%*%Nig.factors)

## Scale factor scores by dividing by factor loading sums to get scores between -2 and 2

loadSum <- colSums(data.frame(Nig.factors))
factor.scores <- apply(factor.scores,1,function(x)x/loadSum)
factor.scores <- data.frame(t(factor.scores))

write.table(factor.scores,"C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/FactorScores_10.csv",sep=",",col.names=TRUE,row.names=FALSE,quote=TRUE,na="NA")
```

III. Recode Response Variables

```
## Code for recoding response variables

library(car) # package for recoding

demoVar <- read.csv("C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/Rec_Imp_10.csv",header=TRUE)

## Questions to add in the model and corresponding recoding

Actor <- as.factor(demoVar[,"q47"])
Safety <- demoVar[,"d23"]
Goals <- demoVar[,"q6"]
Services <- demoVar[,"q7"]
Equality <- demoVar[,"q10"]

## Combine the data sets into initial states for modeling

model.data <- na.omit(data.frame(cbind(factor.scores,Safety,Goals,Services,Equality)))
```

IV. Model Building

```
#### Function to iterate regression models IOT pick the best ones
```

```
library(MASS)
data.best <-
data.frame(matrix(rep(0,nrow(model.data)*ncol(model.data)),nrow(model.data),ncol(model.data)))
names(data.best) <- names(model.data)
for (i in 1:ncol(model.data)){
        reg <- lm(model.data[,i] ~ .,data=model.data[,-c(i)])
        reg.step <- stepAIC(reg,scope = list(upper = ~ ., lower = ~ 1),trace=FALSE)
                if (summary(reg.step)$adj.r.squared > 0.39){
                        data.best[,i] <- model.data[,i]
        }
}

which(colSums(data.best)!=0)

### Building, initializing,& predicting future Issue Stance Scores

## Model Build

rx2 <- lm(X2..U.S..Assist.to.Nigeria ~ . - X4..Social...Essential.Services -
X5..Trust.in.Gov.Agencies - X10..Democracy,data=model.data)
rx2.step <- stepAIC(rx2,scope = list(upper = ~ . - X4..Social...Essential.Services -
X5..Trust.in.Gov.Agencies - X10..Democracy, lower = ~ 1),trace=FALSE)
summary(rx2.step)

rx4 <- lm(X4..Social...Essential.Services ~ . - X2..U.S..Assist.to.Nigeria -
X5..Trust.in.Gov.Agencies - X10..Democracy,data=model.data)
rx4.step <- stepAIC(rx4,scope = list(upper = ~ . - X2..U.S..Assist.to.Nigeria -
X5..Trust.in.Gov.Agencies - X10..Democracy, lower = ~ 1),trace=FALSE)
summary(rx4.step)

rx5 <- lm(X5..Trust.in.Gov.Agencies ~ . - X2..U.S..Assist.to.Nigeria -
X4..Social...Essential.Services - X10..Democracy,data=model.data)
rx5.step <- stepAIC(rx5,scope = list(upper = ~ . - X2..U.S..Assist.to.Nigeria -
X4..Social...Essential.Services - X10..Democracy, lower = ~ 1),trace=FALSE)
summary(rx5.step)

rx10 <- lm(X10..Democracy ~ . - X2..U.S..Assist.to.Nigeria - X4..Social...Essential.Services
- X5..Trust.in.Gov.Agencies,data=model.data)
rx10.step <- stepAIC(rx10,scope = list(upper = ~ . - X2..U.S..Assist.to.Nigeria -
X4..Social...Essential.Services - X5..Trust.in.Gov.Agencies, lower = ~ 1),trace=FALSE)
summary(rx10.step)

## Generate initial Issue Stance Scores using mean factor scores

intx2 <- intersect(names(coef(rx2.step)),names(model.data))
intx4 <- intersect(names(coef(rx4.step)),names(model.data))
intx5 <- intersect(names(coef(rx5.step)),names(model.data))
intx10 <- intersect(names(coef(rx10.step)),names(model.data))
ndx2 <- data.frame(matrix(round(colMeans(model.data[,c(intx2)]),3),1,NROW(intx2),byrow=TRUE))
names(ndx2) <- c(intx2)
ndx4 <- data.frame(matrix(round(colMeans(model.data[,c(intx4)]),3),1,NROW(intx4),byrow=TRUE))
names(ndx4) <- c(intx4)
ndx5 <- data.frame(matrix(round(colMeans(model.data[,c(intx5)]),3),1,NROW(intx5),byrow=TRUE))
names(ndx5) <- c(intx5)
ndx10 <- data.frame(matrix(round(colMeans(model.data[,c(intx10)]),3),1,NROW(intx10),byrow=TRUE))
names(ndx10) <- c(intx10)

## Predict inital Issue Stance Scores

nx2 <- data.frame(round(predict(rx2.step,ndx2,type="response"),3))
nx4 <- data.frame(round(predict(rx4.step,ndx4,type="response"),3))
nx5 <- data.frame(round(predict(rx5.step,ndx5,type="response"),3))
nx10 <- data.frame(round(predict(rx10.step,ndx10,type="response"),3))

## Output initial Issue Stance Score files to excel

library(xlsx)
names(nx2) <- c("X2_Predict")
names(nx4) <- c("X4_Predict")
```

```
names(nx5) <- c("X5_Predict")
names(nx10) <- c("X10_Predict")
write.xlsx(nx2,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X2_Initial_Issue",row.names=FALSE,append=TRUE)
write.xlsx(nx4,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X4_Initial_Issue",row.names=FALSE,append=TRUE)
write.xlsx(nx5,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X5_Initial_Issue",row.names=FALSE,append=TRUE)
write.xlsx(nx10,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X10_Initial_Issue",row.names=FALSE,append=TRUE)

## Read-in SME input files

pdx2 <- read.xlsx("C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/SME.xlsx",sheetIndex=1,sheetName="X2",as.data.frame=TRUE,header=TRUE,keepFormulas=FALSE)
pdx2 <- pdx2[,-c(1,2)]
pdx4 <- read.xlsx("C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/SME.xlsx",sheetIndex=2,sheetName="X4",as.data.frame=TRUE,header=TRUE,keepFormulas=FALSE)
pdx4 <- pdx4[,-c(1,2)]
pdx5 <- read.xlsx("C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/SME.xlsx",sheetIndex=3,sheetName="X5",as.data.frame=TRUE,header=TRUE,keepFormulas=FALSE)
pdx5 <- pdx5[,-c(1,2)]
pdx10 <- read.xlsx("C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/SME.xlsx",sheetIndex=4,sheetName="X10",as.data.frame=TRUE,header=TRUE,keepFormulas=FALSE)
pdx10 <- pdx10[,-c(1,2)]

## Predict future Issue Stance Scores based on events
event <- c(1:20)
p2 <- data.frame(round(predict(rx2.step,pdx2,type="response"),3))
p4 <- data.frame(round(predict(rx4.step,pdx4,type="response"),3))
p5 <- data.frame(round(predict(rx5.step,pdx5,type="response"),3))
p10 <- data.frame(round(predict(rx10.step,pdx10,type="response"),3))

px2 <- cbind(event,p2)
px4 <- cbind(event,p4)
px5 <- cbind(event,p5)
px10 <- cbind(event,p10)

## Output predicted Issue Stance Score files to excel

names(px2) <- c("Event","X2_Predict")
names(px4) <- c("Event","X4_Predict")
names(px5) <- c("Event","X5_Predict")
names(px10) <- c("Event","X10_Predict")
write.xlsx(px2,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X2_Predict_Issue",row.names=FALSE,append=TRUE)
write.xlsx(px4,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X4_Predict_Issue",row.names=FALSE,append=TRUE)
write.xlsx(px5,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X5_Predict_Issue",row.names=FALSE,append=TRUE)
write.xlsx(px10,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="X10_Predict_Issue",row.names=FALSE,append=TRUE)

### Building, initializing, and predicting future OABs

## Model Build

library(mlogit)

wr.data <- data.frame(cbind(Actor,factor.scores))
wr.data <- wr.data[,c(1,3,5,6,11)]

WR <- mlogit.data(wr.data,varying=NULL,choice="Actor",shape="wide")

weight.reg <- mlogit(Actor ~ 1 | X2..U.S..Assist.to.Nigeria + X4..Social...Essential.Services +
X5..Trust.in.Gov.Agencies + X10..Democracy,data=WR,reflevel="0")
wsum <- summary(weight.reg)

## Predict Initial OAB Probabilities
```

```
oab.data <- wr.data[,-c(1)]
wr <- data.frame(matrix(round(colMeans(oab.data),3),1,4,byrow=TRUE))
names(wr) <- names(oab.data)

log0 <- rep(0,1)
log1 <- wsum$coef[["1:(intercept)"]] +
wsum$coef[["1:X2..U.S..Assist.to.Nigeria"]]*wr$X2..U.S..Assist.to.Nigeria +
wsum$coef[["1:X4..Social...Essential.Services"]]*wr$X4..Social...Essential.Services +
wsum$coef[["1:X5..Trust.in.Gov.Agencies"]]*wr$X5..Trust.in.Gov.Agencies +
wsum$coef[["1:X10..Democracy"]]*wr$X10..Democracy
log2 <- wsum$coef[["2:(intercept)"]] +
wsum$coef[["2:X2..U.S..Assist.to.Nigeria"]]*wr$X2..U.S..Assist.to.Nigeria +
wsum$coef[["2:X4..Social...Essential.Services"]]*wr$X4..Social...Essential.Services +
wsum$coef[["2:X5..Trust.in.Gov.Agencies"]]*wr$X5..Trust.in.Gov.Agencies +
wsum$coef[["2:X10..Democracy"]]*wr$X10..Democracy
log3 <- wsum$coef[["3:(intercept)"]] +
wsum$coef[["3:X2..U.S..Assist.to.Nigeria"]]*wr$X2..U.S..Assist.to.Nigeria +
wsum$coef[["3:X4..Social...Essential.Services"]]*wr$X4..Social...Essential.Services +
wsum$coef[["3:X5..Trust.in.Gov.Agencies"]]*wr$X5..Trust.in.Gov.Agencies +
wsum$coef[["3:X10..Democracy"]]*wr$X10..Democracy
log4 <- wsum$coef[["4:(intercept)"]] +
wsum$coef[["4:X2..U.S..Assist.to.Nigeria"]]*wr$X2..U.S..Assist.to.Nigeria +
wsum$coef[["4:X4..Social...Essential.Services"]]*wr$X4..Social...Essential.Services +
wsum$coef[["4:X5..Trust.in.Gov.Agencies"]]*wr$X5..Trust.in.Gov.Agencies +
wsum$coef[["4:X10..Democracy"]]*wr$X10..Democracy
log5 <- wsum$coef[["5:(intercept)"]] +
wsum$coef[["5:X2..U.S..Assist.to.Nigeria"]]*wr$X2..U.S..Assist.to.Nigeria +
wsum$coef[["5:X4..Social...Essential.Services"]]*wr$X4..Social...Essential.Services +
wsum$coef[["5:X5..Trust.in.Gov.Agencies"]]*wr$X5..Trust.in.Gov.Agencies +
wsum$coef[["5:X10..Democracy"]]*wr$X10..Democracy

logits <- cbind(log0,log1,log2,log3,log4,log5)
prob <- data.frame(round(exp(logits)/rowSums(exp(logits)),3)) # This is the data frame of
probabilities
colnames(prob) <-
c("Rebel_Groups_Predict","International_Terrorists_Predict","Common_Criminals_Predict",
"Military_Predict","Government_Predict","Foreign_Countries_Predict")

## Output initial OAB Probability files to excel

names(prob[1]) <- c("Rebel_Groups_Predict")
names(prob[2]) <- c("International_Terrorists_Predict")
names(prob[3]) <- c("Common_Criminals_Predict")
names(prob[4]) <- c("Military_Predict")
names(prob[5]) <- c("Government_Predict")
names(prob[6]) <- c("Foreign_Countries_Predict")
write.xlsx(prob[1],file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Rebels_Initial_OAB",row.names=FALSE,append=TRUE)
write.xlsx(prob[2],file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Terrorists_Initial_OAB",row.names=FALSE,append=TRUE)
write.xlsx(prob[3],file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Criminals_Initial_OAB",row.names=FALSE,append=TRUE)
write.xlsx(prob[4],file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Military_Initial_OAB",row.names=FALSE,append=TRUE)
write.xlsx(prob[5],file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Government_Initial_OAB",row.names=FALSE,append=TRUE)
write.xlsx(prob[6],file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="ForiegnCountries_Initial_OAB",row.names=FALSE,append=TRUE)

## Predict future OAB Probabilities based on events

pd <- cbind(px2,px4,px5,px10)[,c(2,4,6,8)]

log00 <- rep(0,20)
log11 <- wsum$coef[["1:(intercept)"]] + wsum$coef[["1:X2..U.S..Assist.to.Nigeria"]]*pd$X2_Predict
+ wsum$coef[["1:X4..Social...Essential.Services"]]*pd$X4_Predict +
wsum$coef[["1:X5..Trust.in.Gov.Agencies"]]*pd$X5_Predict +
wsum$coef[["1:X10..Democracy"]]*pd$X10_Predict
```

```
log22 <- wsum$coef[["2:(intercept)"]] + wsum$coef[["2:X2..U.S..Assist.to.Nigeria"]]*pd$X2_Predict
+ wsum$coef[["2:X4..Social...Essential.Services"]]*pd$X4_Predict +
wsum$coef[["2:X5..Trust.in.Gov.Agencies"]]*pd$X5_Predict +
wsum$coef[["2:X10..Democracy"]]*pd$X10_Predict
log33 <- wsum$coef[["3:(intercept)"]] + wsum$coef[["3:X2..U.S..Assist.to.Nigeria"]]*pd$X2_Predict
+ wsum$coef[["3:X4..Social...Essential.Services"]]*pd$X4_Predict +
wsum$coef[["3:X5..Trust.in.Gov.Agencies"]]*pd$X5_Predict +
wsum$coef[["3:X10..Democracy"]]*pd$X10_Predict
log44 <- wsum$coef[["4:(intercept)"]] + wsum$coef[["4:X2..U.S..Assist.to.Nigeria"]]*pd$X2_Predict
+ wsum$coef[["4:X4..Social...Essential.Services"]]*pd$X4_Predict +
wsum$coef[["4:X5..Trust.in.Gov.Agencies"]]*pd$X5_Predict +
wsum$coef[["4:X10..Democracy"]]*pd$X10_Predict
log55 <- wsum$coef[["5:(intercept)"]] + wsum$coef[["5:X2..U.S..Assist.to.Nigeria"]]*pd$X2_Predict
+ wsum$coef[["5:X4..Social...Essential.Services"]]*pd$X4_Predict +
wsum$coef[["5:X5..Trust.in.Gov.Agencies"]]*pd$X5_Predict +
wsum$coef[["5:X10..Democracy"]]*pd$X10_Predict

logits1 <- cbind(log00,log11,log22,log33,log44,log55)
prob1 <- data.frame(round(exp(logits1)/rowSums(exp(logits1)),3))
colnames(prob1) <- c("Rebel Groups","International Terrorists","Common
Criminals","Military","Government","Foreign Countries")

## Output predicted OAB Probability files to excel

poab0 <- data.frame(cbind(event,prob1[,1]))
poab1 <- data.frame(cbind(event,prob1[,2]))
poab2 <- data.frame(cbind(event,prob1[,3]))
poab3 <- data.frame(cbind(event,prob1[,4]))
poab4 <- data.frame(cbind(event,prob1[,5]))
poab5 <- data.frame(cbind(event,prob1[,6]))
names(poab0) <- c("Event","Rebel_Groups_Predict")
names(poab1) <- c("Event","International_Terrorists_Predict")
names(poab2) <- c("Event","Common_Criminals_Predict")
names(poab3) <- c("Event","Military_Predict")
names(poab4) <- c("Event","Government_Predict")
names(poab5) <- c("Event","Foreign_Countries_Predict")
write.xlsx(poab0,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Rebels_Predict_OAB",row.names=FALSE,append=TRUE)
write.xlsx(poab1,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Terrorists_Predict_OAB",row.names=FALSE,append=TRUE)
write.xlsx(poab2,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Criminals_Predict_OAB",row.names=FALSE,append=TRUE)
write.xlsx(poab3,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Military_Predict_OAB",row.names=FALSE,append=TRUE)
write.xlsx(poab4,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="Government_Predict_OAB",row.names=FALSE,append=TRUE)
write.xlsx(poab5,file="C:/Users/tmdevean/Desktop/IW TWG/2010 Sahel
Survey/ALL.xlsx",sheetName="ForeignCountries_Predict_OAB",row.names=FALSE,append=TRUE)
```

V. Use Case

```
### Example Use Case

time.step <- data.frame(c(1:200))
names(time.step) <- c("Time")
events <- data.frame(sample(1:20,200,replace=T))
names(events) <- c("Event")

event.list1 <- merge(cbind(time.step,events),px2)
event.list2 <- merge(cbind(time.step,events),px4)
event.list3 <- merge(cbind(time.step,events),px5)
event.list4 <- merge(cbind(time.step,events),px10)
event.list5 <- merge(cbind(time.step,events),poab0)
event.list6 <- merge(cbind(time.step,events),poab1)
event.list7 <- merge(cbind(time.step,events),poab2)
event.list8 <- merge(cbind(time.step,events),poab3)
event.list9 <- merge(cbind(time.step,events),poab4)
event.list10 <- merge(cbind(time.step,events),poab5)
```

```
event.list <-
cbind(event.list1,event.list2,event.list3,event.list4,event.list5,event.list6,event.list7,event.l
ist8,event.list9,event.list10)

event.list <- event.list[,c(1,2,3,6,9,12,15,18,21,24,27,30)]

event.list <- event.list[order(event.list[,"Time"]),]
event.list <- event.list[,c(2,1,3,4,5,6,7,8,9,10,11,12)]

in.time <- data.frame(c(0))
names(in.time) <- c("Time")
in.event <- data.frame(c(0))
names(in.event) <- c("Event")

event.list <-
rbind(cbind(in.time,in.event,nx2,nx4,nx5,nx10,prob[1],prob[2],prob[3],prob[4],prob[5],prob[6]),ev
ent.list)

## Issue Stance Score Plots
par(mfrow=c(2,2))
plot(event.list$Time,event.list$X2_Predict,type="l",xlab="Time Step",ylim=c(-2,2),ylab="Issue
Stance Score",main="'U.S. Assistance to Nigeria' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$X2_Predict,iter=10),lty="dashed",col="139")

plot(event.list$Time,event.list$X4_Predict,type="l",xlab="Time Step",ylim=c(-2,2),ylab="Issue
Stance Score",main="'Social & Essential Services' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$X4_Predict,iter=10),lty="dashed",col="139")

plot(event.list$Time,event.list$X5_Predict,type="l",xlab="Time Step",ylim=c(-2,2),ylab="Issue
Stance Score",main="'Trust in Government Agencies' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$X5_Predict,iter=10),lty="dashed",col="139")

plot(event.list$Time,event.list$X10_Predict,type="l",xlab="Time Step",ylim=c(-2,2),ylab="Issue
Stance Score",main="'Democracy' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$X10_Predict,iter=10),lty="dashed",col="139")

## OAB Probability Plots
par(mfrow=c(2,3))
plot(event.list$Time,event.list$Rebel_Groups_Predict,type="l",xlab="Time
Step",ylim=c(0.05,0.15),ylab="Probability",main="'Rebel Groups' OAB Probability over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$Rebel_Groups_Predict,iter=10),lty="dashed",col="139")

plot(event.list$Time,event.list$International_Terrorists_Predict,type="l",xlab="Time
Step",ylim=c(0,0.1),ylab="Probability",main="'International Terrorists' OAB Probability over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$International_Terrorists_Predict,iter=10),lty="dashed",co
l="139")

plot(event.list$Time,event.list$Common_Criminals_Predict,type="l",xlab="Time
Step",ylim=c(0.1,0.3),ylab="Probability",main="'Common Criminals' OAB Probability over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$Common_Criminals_Predict,iter=10),lty="dashed",col="139")

plot(event.list$Time,event.list$Military_Predict,type="l",xlab="Time
Step",ylim=c(0.05,0.1),ylab="Probability",main="'Military' OAB Probability over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$Military_Predict,iter=10),lty="dashed",col="139")

plot(event.list$Time,event.list$Government_Predict,type="l",xlab="Time
Step",ylim=c(0.4,0.7),ylab="Probability",main="'Government' OAB Probability over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$Government_Predict,iter=10),lty="dashed",col="139")
```

```
plot(event.list$Time,event.list$Foreign_Countries_Predict,type="l",xlab="Time
Step",ylim=c(0.01,0.04),ylab="Probability",main="'Foreign Countries' OAB Probability over
Time",col="2",col.main="4",font.lab="2",font.main="2")
lines(lowess(event.list$Time,event.list$Foreign_Countries_Predict,iter=10),lty="dashed",col="139"
)

sh.elist <- event.list[,-c(1,2)]
delta.event <- cumsum(sh.elist)
delta.event <- cbind(data.frame(c(1:201)),delta.event)
names(delta.event) <-
c("Time","X2_Delta","X4_Delta","X5_Delta","X10_Delta","Rebel_Delta","Terrorist_Delta",
"Criminal_Delta","Military_Delta","Government_Delta","Foreign_Delta")

## Issue Stance Score Cumulative Plots
par(mfrow=c(2,2))
plot(delta.event$Time,delta.event$X2_Delta,type="l",xlab="Time Step",ylab="Issue Stance Score
Delta",main="Change in 'U.S. Assistance to Nigeria' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")

plot(delta.event$Time,delta.event$X4_Delta,type="l",xlab="Time Step",ylab="Issue Stance Score
Delta",main="Change in 'Social & Essential Services' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")

plot(delta.event$Time,delta.event$X5_Delta,type="l",xlab="Time Step",ylab="Issue Stance Score
Delta",main="Change in 'Trust in Government Agencies' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")

plot(delta.event$Time,delta.event$X10_Delta,type="l",xlab="Time Step",ylab="Issue Stance Score
Delta",main="Change in 'Democracy' Issue Stance Score over
Time",col="2",col.main="4",font.lab="2",font.main="2")
```

LIST OF REFERENCES

Agresti, A. (1996). *An Introduction to Categorical Data Analysis* (1st ed., p. 206). John Wiley & Sons, Inc.

Costello, A. B., & Osborne, J. W. (2005). Best Practices in Exploratory Factor Analysis: Four Recommendations for Getting the Most From Your Analysis. *Practical Assessment, Research and Evaluation*, *10*(7), 1-9.

DeCoster, J. (1998). Overview of Factor Analysis. Retrieved July 25, 2012, from http://www.stat-help.com/notes.html

Dziedzic, M., Sotirin, B., & Agoglia, J. (Eds.). (2008). *Measuring Progress in Conflict Enviroments (MPICE)* (1st ed.).

D'Orazio, M. (2011). StatMatch: Statistical Matching.

Harman, H. H. (1976). *Modern Factor Analysis* (3rd ed.). University of Chicago Press.

Hayton, J. C., Allen, D. G., & Scarpello, V. (2004). Factor Retention Decisions in Exploratory Factor Analysis: A Tutorial on Parallel Analysis. *Organizational Research Methods*, *7*(2), 191-205.

Hosmer, D. W., & Lemeshow, S. (2000). *Applied Logistic Regression* (2nd ed., pp. 260-264). John Wiley & Sons, Inc.

Johnson, R. A., & Wichern, D. W. (2002). *Applied Multivariate Statistical Analysis* (5th ed., p. 504). Prentice-Hall.

Kilcullen, D. J. (2009). Measuring Progress in Afghanistan.

Kline, P. (1994). *An Easy Guide to Factor Analysis* (1st ed.). Routledge.

Kulzy, W. W. (2012). *Modeling Indigenous Population Attitudes in Support of Irregular Warfare Analysis*. Naval Postgraduate School.

Luce, R. D., & Tukey, J. W. (1964). Simultaneous Conjoint Measurement: A New Scale of Fundamental Measurement. *Journal of Mathematical Psychology*, *1*, 1-27.

Mulaik, S. A. (2009). *Foundations of Factor Analysis* (2nd ed., p. 136). CRC Press.

Obama, B. (2009). Speech by President Barack Obama to the Ghanian Parliament.

Ploch, L. (2011). Africa Command: U.S. Strategic Interests and the Role of the U.S. Military in Africa. *CRS Report for Congress*. Retrieved July 24, 2012, from http://www.fas.org/sgp/crs/natsec/RL34003.pdf

Revelle, W. (2011). psych: Procedures for Psychological, Psychometric, and Personality Research.

The White House. (2002). *The National Security Strategy of the United States of America*. Washington, DC.

United States. (2006). *Counterinsurgency: Field Manual 3-24*. Washington, DC: Headquarters, Dept. of the Army.

United States. (2009). *Tactics in Counterinsurgency: Field Manual 3-24.2*. Washington, DC: Headquarters, Dept. of the Army.

Upshur, W. P., Roginski, J. W., & Kilcullen, D. J. (2012). Recognizing Systems in Afghanistan. *Prism*, *3*(3), 87-104.

Varner, M. A. (2007). The Strategic Importance of Africa Command. Retrieved June 29, 2012, from http://www.thepresidency.org/storage/documents/Vater/Varner.pdf

APPENDIX K. CHANGES TO THE USER GUIDE FOR SIM 3.0

This Appendix highlights the sections of the SIM 2.0 User Guide that changed for SIM 3.0. There are no new worksheets in the scenario file. SIM 3.0 eliminated the need for the following worksheets:

- BeliefPrototype
- BeliefPositionPrototype
- IssuePositionPrototype
- AttitudePositionPrototype
- AgentBeliefs
- CaseFiles
- BayesNetFiles
- AgentNets

The following list outlines the changes in this scenario file worksheets:

- IssuePrototype - 2 columns added: minSupportStrength and maxSupportStrength
- AttitudePrototype - 2 columns added: minSupportStrength and maxSupportStrength
- CognitiveArchitecture - 1column removed: effectsLambda
- IssueSatisfcationType - 1 column removed: position
- AgentIssues - 1 column added: initialValueDistribution
- AgentAttitudes - 1 column added: initialValueDistribution
- UtilityIssues - 1 column removed: position
- AgentBehaviors - 2 columns removed: initialCaseFile and weight
- ScriptedEffects - contains 5 columns: index, issuePrototype, initiator, receiver and effectValue
- ScriptedAttitudeEffects - contains 5 columns: index, attitudePrototype, initiator, receiver and effectValue
- IssueActionEffects - contains 7 columns: index, issuePrototype, receiver, consumableType, providerAssociation, outcome and effectValue

- AttitudeActionEffects - contains 7 columns: index, attitudePrototype, receiver, consumableType, providerAssociation, outcome and effectValue
- PaveInterface - 1 column removed: issuePosition

The following pages from the SIM 3.0 User Guide show the changes. The complete document resides in the "doc" folder of the source code for SIM 3.0.

Social Impact Module (SIM) Version 3 User Manual

TRADOC Analysis Center

29 September 2012

significant and must match the names that appear in Tables 1 through 48. The data are entered in 48 worksheets/tables and are described as follows:

7.3.1 ScenarioData Worksheet/Table

Provides information for controlling the run. Only one row of data is required. The data is described in Table 7.3.1.

Column/Field Name	Type	Description
ScenarioLength	double	The length of the scenario in arbitrary time units.
Replications	int	The number of replications to run.
verbose	String	Indicates whether the event list should be shown after each event is executed. Used only for debugging, therefore “FALSE” should normally be entered.
reallyVerbose	String	Indicates whether additional debug/trace information should be shown. Used only for debugging, therefore “FALSE” should normally be entered.

Table 7.3.1 ScenarioData Worksheet/Table.

7.3.2 Seeds Worksheet/Table

Specifies the seeds for the random number generator for each replication. One row must be filled out for each replication as shown in Table 7.3.2.

Column/Field Name	Type	Description
replication	int	The replication number.
seed	long	The seed value for the replication.

Table 7.3.2 Seeds Worksheet/Table.

7.3.3 IssuePrototype Worksheet/Table

Defines the issues important to the agents based on the scenario. These are called IssuePrototypes and each row of data defines an IssuePrototype as shown in Table 7.3.3.

Column/Field Name	Type	Description
name	String	The name of the issue.
shortDescripiton	String	A brief description of this issue.
longDescription	String	A more detailed description of this issue.
minSupportStrength	double	The upper bound of the strength of support for this issue. It indicates the agent's strongest possible support for this issue.
maxSupportStrength	double	The lower bound of the strength of support for this issue. It indicates the agent's least possible support, or no support, for this issue.

Table 7.3.3 IssuePrototype Worksheet/Table.

7.3.4 AttitudePrototype Worksheet/Table

An AttitudePrototype defines an attitude that population agents display towards an AgentPrototype representing an external player (see 7.3.11 "AgentPrototype Worksheet/Table"). Examples of external players are coalition forces, the host nation government, NGOs, mass media, and insurgents. Each row of data defines an AttitudePrototype as shown in Table 7.3.4.

Column/Field Name	Type	Description
name	String	The name of the attitude.
attitudeTowards	String	The name of the AgentPrototype defined in Table 7.3.11 that this attitude is directed towards. (See Note.)
minSupportStrength	double	The upper bound of the range of the values that quantify this attitude. It represents the most positive sentiment towards the AgentPrototype that this attitude directed towards.
maxSupportStrength	double	The lower bound of the range of the values that quantify this attitude. It represents the most negative sentiment towards the AgentPrototype that this attitude directed towards.

Table 7.3.4 AttitudePrototype Worksheet/Table.

Note: The "isExternal" field of the AgentPrototype (see Table 7.3.11) should be "TRUE".

7.3.5 SocialDimension Worksheet/Table

Defines the social dimensions over which link weights for each agent pair will be calculated. Each row of data defines a SocialDimension as shown in Table 7.3.5.

The social dimension may be either a static ascribed characteristic such as tribe, education, political affiliation or age; or it may be a belief, value, interest or issue in which the stances/positions may change during a simulation run. Although in reality the static dimensions may move over time, SIM considers them fixed throughout a run.

Each dimension must be assigned a weight that indicates the relative importance of that dimension. If d dimensions are defined in this table and c_i is the weight assigned to dimension i, each c_i must obey the following constraints: $0 \le c_i \le 1 \forall i \in (1,\ldots,d)$ and

$$\sum_{i=1}^{d} c_i = 1.$$

Column/Field Name	Type	Description
name	String	The name of the social dimension. If this dimension is dynamic, the name must be that of a BeliefPrototype declared in Table 7.3.3 or an

		IssuePrototype declared in Table 7.3.3.
homophilyWeight	double	The relative importance of this dimension in the range [0.0, 1.0]. The sum of these weights over all of the dimensions must add to 1.0.

Table 7.3.5 SocialDimension Worksheet/Table

7.3.6 SocialDimensionValueType Worksheet/Table

Defines the classifications/categories within each social dimension and assigns a numerical value to each one. The numerical value defines the position that that category occupies along the dimension. Each row of data defines a category within a SocialDimension as shown in Table 7.3.6. Note that category names need to be unique within a social dimension but may be used repeatedly between different social dimensions.

Column/Field Name	Type	Description
category	String	The name of the classification/category of a static SocialDimension or the position/stance of a BeliefPrototype or IssuePrototype.
socialDimension	String	The name of a SocialDimension defined in Table 7.3.5 that category is assigned to.
value	double	The value must be greater than or equal to 0.

Table 7.3.6 SocialDimensionValueType Worksheet/Table.

The values assigned to each category in a given dimension are used to determine the distance between agents in that dimension. The values must be on the same scale within a dimension, but different dimensions may use different scales. For example, a five-point Likert scale may be used on one dimension while a 0-100 socioeconomic index may be used on another dimension. The distances between agents in a given dimension will be normalized by the maximum distance between any two agents in that dimension, resulting in a scalar between 0 and 1. Once the distances have been normalized for each dimension, they can be combined to calculate the social distance between two agents which, in turn, can be used to calculate the link weight of the agent pair[4].

Example: Suppose there is a dimension called “Disposition” that classifies agents in the civilian population as either “Rural” or “Urban”. Suppose Rural is assigned a value of 1 and Urban is assigned a value of 2. The distance between a Rural agent and an Urban agent will be 1 within this dimension while the distance between two Rural agents or two Urban agents will be 0.

7.3.7 CognitiveArchitecture Worksheet/Table

A single row of data must be entered that covers an assortment of parameters required by the Cognitive Architecture. The entries are described in Table 7.3.7.

Column/Field Name	Type	Description

[4] S. Lieberman, “Some Next Steps for Social Networks in the Cultural Geography Model”, working paper dated 2009 Sep 01

selectiveAttentionThreshold	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the attention threshold for each agent. (See Note 1.)
workingMemoryCapacity	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the working memory capacity for each agent. (See Note 2.)
expectedCommunication	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the expected number of times an agent (from the civilian population) will communicate with another civilian agent over a set time period, specified by *expectedCommunicationTimeUnits*. (See Notes 2 and 3.)
expectedCommunicationTimeUnits	double	The time period over which the number of times each civilian agent communicates is tracked.
experienceThreshold	int	Determines whether an agent has enough experience. (See Note 4.)
volatilityThreshold	double	The threshold that indicates, based on recent actions, whether an agent has tended to select actions that result in consistent rewards over actions that result in uneven (volatile) rewards. (See Note 5.)
volatilityPeriods	int	The number of time periods over which volatility is measured. (See Note 5.)
volatilityPeriodLength	double	The length of each time period over which volatility is measured (See Note 5.)
physiologicalWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0, 1.0]. This weight is applied to the motivation score for immediate physiological needs. (See Note 6.)
selfProtectionWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0,

		1.0]. This weigh is applied to the motivation score for self-protection. (See Note 6.)
affiliationWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0, 1.0]. This weight is applied to the motivation score for affiliation. (See Note 6.)
statusEsteemWeight	double	Weight applied to motivation scores for calculating satisfaction in the range [0.0, 1.0]. This weight is applied to the motivation score for status/esteem. (See Note 6.)
temperature	double	The temperature for generating the Boltzmann distribution in the range [0.0, ∞). Used during the MetaCognition process to determine goals.
filterTrust	String	Enter "TRUE" or "FALSE" if trust filtering will be on or off, respectively. (See Note 7.)

Table 7.3.7 CognitiveArchitecture Worksheet/Table.

Note 1: The attention threshold is used to determine whether a percept should be added to working memory. If the age of the percept (time percept received minus time percept was formed) is less than the attention threshold, the percept is added to working memory; otherwise, it is discarded. SIM will throw a RuntimeException if the distribution generates a value that is less than or equal to zero.

Note 2: Since the simkit.random.RandomVariate.generate() method returns a double, the result will be rounded to the nearest integer. If the value after rounding is less than zero, SIM will throw a RuntimeException. If the value is zero, the meta-cognition process will handle this by setting the agent's motivation score for sending communication to zero. The agent will still be allowed to receive communication sent by another agent.

Note 3: Currently communication is only considered between agents in the civilian population. Informant communication (where a civilian agent provides information to an agent representing an external player, such as a coalition force agent) is being considered for future implementation.

Note 4: Experience is defined by the number of trials of each action taken. The agent tracks the number of times each action was taken. If the number of times action X was taken is less than or equal to the value held in the "experienceThreshold" field, the agent is considered to have insufficient experience with regard to Action X; otherwise, the agent is considered to have sufficient experience. Experience is one of two factors used during the ActionSelection process to choose a decision method: exploration learning, recognition prime decision making (RPD), or mental stimulation.

Note 5: Volatility is a measure of risk. It is the second of two factors used during the ActionSelection process to choose a decision method. Volatility is measured over a set number of time periods (“volatilityPeriods”) of specified length (“volatilityPeriodLength”). For each action taken over these time periods, the maximum and minimum expected utilities resulting from these actions are tracked. For a given action in a given time period, the ratio of the maximum expected utility over the minimum expected utility yields the volatility of that action in that time period. The maximum volatility is the maximum volatility over all actions and all time periods. If the maximum volatility exceeds a specified threshold (“volatilityThreshold”), the volatility is considered high (more risk); otherwise the volatility is considered low (less risk).

Note 6: The sum of “physiologicalWeight”, “selfProtectionWeight”, “affiliationWeight” and “statusEsteemWeight” must add to 1.0.

Note 7: If trust filtering is on, an agent will only communicate with the agents in its social network that it trusts, and an agent that receives information from another agent will accept that information only if it trusts the sender; otherwise, the information is ignored. If trust filtering is off, an agent will always communicate with the agents in its social network, and it will always accept information that it receives from the other agents in its social network.

7.3.8 IssueSatisfactionType Worksheet/Table

During the MetaCognition process an agent performs a cognitive appraisal where a satisfaction value in the range [0.0, 1.0] is calculated. The larger this value, the more “satisfied” the agent is with the current state of affairs. The calculation of satisfaction consists of two components: motivation scores and issue stances. This table identifies the issue(s) that will contribute to the calculation.

Column/Field Name	Type	Description
name	String	The name for grouping the issues. This name will be referenced by an AgentPrototype listed in Table 7.3.11.
issue	String	The name of the IssuePrototype defined in Table 7.3.3. This is the issue that is considered relevant for calculating satisfaction.
weight	double	The weight that *issue* contributes in the range [0.0, 1.0]. The weights of all issues assigned to *name* must sum to 1.0.

Table 7.3.8 IssueSatisfactionType Worksheet/Table

Each group of issues identified by the “name” field can be tailored to one or more AgentPrototypes (see 7.3.11 “AgentPrototype Worksheet/Table”). In this way, one set of issues and positions can be created that are appropriate for, say, “passive” agents while another set of issues and positions can be created for agents that are “radical”.

7.3.9 PerceptUmpire Worksheet/Table

Defines the PerceptUmpire. Only one row of data is required. The data is described in Table 7.3.9.

Column/Field Name	Type	Description
name	String	The name of the umpire.
class	String	The class name of a PerceptUmpire defined in the rucg.mas.behavior.cognitive package. (See Note.)

Table 7.3.9 PerceptUmpire Worksheet/Table

Note: Enter either “CgPerceptUmpire” or “rucg.mas.behavior.cognitive.CgPerceptUmpire” (without the quotes).

7.3.10 ConsumableType Worksheet/Table

Defines the types of goods and services consumed by agents and stored at infrastructures. Each row of data defines a ConsumableType as shown in Table 7.3.10.

Column/Field Name	Type	Description
name	String	The name of the type of consumable.

Table 7.3.10 ConsumableType Worksheet/Table.

7.3.11 AgentPrototype Worksheet/Table

Agents are classified by AgentPrototype. Each row of data defines an AgentPrototype as shown in Table 7.3.11.

Column/Field Name	Type	Description
name	String	The name of the prototype.
isGroup	String	If this prototype represents a group, organization or institution, enter “TRUE”; otherwise, enter “FALSE”.
isExternal	String	If this prototype represents an external player, enter “TRUE”. Enter “FALSE” if this prototype represents a stereotype of the civilian population. (See Note 2.)
isMedia	String	If *isGroup* is “TRUE” and this prototype represents the mass media, enter “TRUE”; otherwise, enter “FALSE”.
moveRate	String	If *isExternal* is “FALSE”, enter the distribution for generating a movement rate when an agent of this type needs to travel to and from an infrastructure to obtain goods or services. (See Notes 1 and 3) Ignored if *isExternal* is “TRUE”.
issueSatisfactionType	String	If *isExternal* is “TRUE”, enter the name of the IssueSatisfactionType defined in Table 7.3.8. If not applicable, enter “NA”. Ignored if *isExternal* is “FALSE”. (See Note 4.)

Table 7.3.11 AgentPrototype Worksheet/Table.

Note 1: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 2: Examples of external players are coalition forces, the host nation government, NGOs, mass media, and insurgents.

Note 3: Movement rate distributions should be entered if the “spatialMethod” field of the SimpleInfrastructureUmpire defined in Table 7.3.46 is “PROXIMITY”. Each distribution should generate a rate measured in kilometers per unit time if coordinates of Locations (see 7.3.24 “Location Worksheet/Table”) are GEODETIC, MILGRID or UTM. If Locations use ARBITRARY_X_Y coordinates, however, the distance is measured in an arbitrary unit consistent with that coordinate system. If the “spatialMethod” field of the SimpleInfrastructureUmpire is “COLLOCATION”, the “delayClass” field in table 7.3.45 should be used instead and “NA” should be entered in the “moveRate” field.

Note 4: When an agent performs a cognitive appraisal it calculates a satisfaction value. This value is in the range [0.0, 1.0] and the larger the value, the more “satisfied” the agent is with the current state of affairs. The calculation of satisfaction consists of two components: motivation scores and issue stances. The “issueSatisfactionType” field addresses the issue stance component and identifies the group of issues that are relevant to calculating the satisfaction. These groups were specified in Table 7.3.8 “IssueSatisfactionType Worksheet/Table”. If “NA” is entered in the field, the satisfaction calculation will only consider motivation scores. Note that external players do not evaluate issues; therefore, they will base their satisfaction on motivation scores only.

7.3.12 AgentSocialDimensions Worksheet/Table

Assigns to each AgentPrototype a category from each static SocialDimension that best characterizes the prototype in that dimension. One row is filled out for each static dimension for each prototype as shown in Table 7.3.12. External agent prototypes only have SocialDimensions if they are participating in the dynamic social network.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.11.
socialDimension	String	The name of a **static** SocialDimension defined in Table 7.3.5.
Category	String	The name of a category that is assigned to *socialDimension* in Table 7.3.6.

Table 7.3.12 AgentSocialDimensions Worksheet/Table.

7.3.13 AgentIssues Worksheet/Table

Defines issues important to AgentPrototype. One row is filled out for each issue important to an AgentPrototype as shown in Table 7.3.13.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.11 whose “isExternal” field is “FALSE”.
issuePrototype	String	The name of the IssuePrototype defined in Table 7.3.3.
initialValueDistribution	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the initial strength of support of *issuePrototype* for all agents of type *agentPrototype*. (See Notes.)

Table 7.3.13 AgentIssues Worksheet/Table.

Note : Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(1.0), Normal(1.0, 0.001), and Triangle(-0.5, 1.0, 0.25). If the value generated by the distribution is less than the “minSupportStrength” of the IssuePrototype defined in Table 7.3.3, the initial strength of support is set to “minSupportStrength”. Likewise, if the value generated is greater than the “maxSupportStrength” of the IssuePrototype, the initial strength of support is set to “maxSupportStrength”.

7.3.14 AgentAttitudes Worksheet/Table

Defines attitudes held by AgentPrototype. One row is filled out for each attitude held by an AgentPrototype as shown in Table 7.3.14.

Column/Field Name	Type	Description
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.11 whose “isExternal” field is “FALSE”.
attitudePrototype	String	The name of the AttitudePrototype defined in Table 7.3.4.
initialValueDistribution	String	The class name of a distribution defined in the simkit.random package or a java class implementing the simkit.random.RandomVariate interface. This distribution is used to generate the initial attitude of *attitudePrototype* for all agents of type *agentPrototype*.

Table 7.3.14 AgentAttitudes Worksheet/Table

Note: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(0.1), Normal(0.5, 0.001), and Triangle(0.0, 1.0, 0.5). If the value generated by the distribution is less than the “minSupportStrength” of the AttitudePrototype defined in Table 7.3.4, the initial attitude is set to “minSupportStrength”. Likewise, if the value generated is greater than the “maxSupportStrength” of the AttitudePrototype, the initial attitude is set to “maxSupportStrength”.

7.3.15 Agent Worksheet/Table

Defines each agent to be instantiated in the scenario. One row of data is entered for each agent as shown in Table 7.3.15.

Column/Field Name	Type	Description
name	String	The name of the agent.
agentPrototype	String	The name of the AgentPrototype defined in Table 7.3.11.
initialLocation	String	Either the name of the Location defined in Table 7.3.24 where this agent will be initially located (i.e., at time 0), or “ANY”. If the latter, the SimpleLocationUmpire will assign an initial Location to this agent.
keyLeader	Boolean	True if this agent is a key leader in the scenario.
trustFraction	double	The fraction of nearest K neighbors this agent will choose to communicate with, in the range [0.0, 1.0]. (See Note 1.)
trustTemperature	double	The temperature at which this agent chooses its trustworthy agents, in the range (0.0, ∞). (See Notes 1 and 2.)
defaultTrust	double	The default (initial) trust value this agent uses when it has no trust value about another agent. (See Note 1.)
lambdaSend	double	(See Note 1.)
gammaOrOneSend	double	(See Note 1.)
exploreModeSend	String	Enter “BOLTZMANN” or “EPSILON_GREEDY” (See Note 1.)
epsilonOrTemperatureSend	double	(See Note 1.)
defaultUtilitySend	double	(See Note 1.)
modeSend	String	Enter “Q_LEARNING”, “SARSA” or “DIRECT_Q_COMPUTATION” (See Note 1.)
lambdaReceive	double	(See Note 1.)
gammaOrOneReceive	double	(See Note 1.)
exploreModeReceive	String	Enter “BOLTZMANN” or “EPSILON_GREEDY” (See Note 1.)
epsilonOrTemperatureReceive	double	(See Note 1.)
defaultUtilityReceive	double	(See Note 1.)
modeReceive	String	Enter “Q_LEARNING”, “SARSA” or

		Boltzmann distribution in the range (0.0, ∞). (See Note 3.)
minTemperature	double	The minimum temperature allowed in the range (0.0, ∞). (See Note 3.)

Table 7.3.19 UtilityBehavior Worksheet/Table

Note 1: The following UtilityBehavior classes are currently implemented:

a. SimpleUtilityBehavior
b. UtilityCommBehavior
c. UtilityInfrastructureTpBehavior

The package name "rucg.mas.behavior.utility" may be optionally prefixed to the class entry. Therefore, "UtilityInfrastructureTpBehavior" and "rucg.mas.behavior.utility.UtilityInfrastructureTpBehavior" are legitimate entries (without the quotes).

Note 2: The values of normWeight, attitudeWeight and controlWeight must sum to 1.0.

Note 3: initialTemperature must be greater than minTemperature.

7.3.20 UtilityIssues Worksheet/Table

Lists the issues evaluated by each UtilityBehavior. A row must be filled out for each issue considered by the UtilityBehavior as shown in Table 7.3.20.

Column/Field Name	Type	Description
utilityBehavior	String	The name of the UtilityBehavior defined in Table 7.3.19
issue	String	The name of the IssuePrototype defined in Table 7.3.3. This is the issue that will be evaluated by *utilityBehavior*.
weight	double	The weight that *issue* contributes in the range [0.0, 1.0]. The weights of all issues considered by *utilityBehavior* must sum to 1.0.

Table 7.3.20 UtilityIssues Worksheet/Table

7.3.21 Method Worksheet/Table

A Method holds a set of method levels where each level determines one or more courses of action available to an agent. A method level provides a means to classify the condition of an agent. For example, an agent's condition may be classified to be one of five levels called "Very Positive", "Positive", "Neutral", "Negative", "Very Negative". These five levels would be grouped into one Method. This worksheet simply defines how the levels are grouped. The mappings from level to courses of action are entered in 7.3.22, "BehaviorMethodAction Worksheet/Table". One row must be filled out for each level as shown in Table 7.3.21.

		in this field and *behavior* must be consistent with the entries in the “behaviorName” and “intentNodeState” fields in Table 7.3.18.

Table 7.3.22 BehaviorMethodAction Worksheet/Table

7.3.23 AgentBehaviors Worksheet/Table

Declares the planned behaviors that each agent will simulate. It sets the initial state of the agent’s behavior, the method the agent uses to select the action it will take, and determines how frequently the behaviors are carried out. Each row is filled out according to Table 7.3.23.

Column/Field Name	Type	Description
agent	String	The name of the agent defined in Table 7.3.15.
behaviorName	String	The name of the behavior declared in Table 7.3.17.
utilityBehavior	String	If applicable, the name of the UtilityBehavior defined in Table 7.3.19.
consumableType	String	If the “behaviorType” field in Table 7.3.17 is “INFRASTRUCTURE” for *behaviorName*, the name of the ConsumableType from Table 7.3.10 that will be restocked.
initialExecuteTime	String	Enter “NONE” or the distribution to generate the (simulation) time that this behavior will be executed for the first time. (See Notes 1 and 2.)
executeInterval	String	Enter “NONE” or the distribution to generate a waiting time before this behavior is repeated. (See Notes 1 and 2.)
stopBehaviorTime	String	Enter the distribution to generate the (simulation) time to stop this behavior. Enter “NONE” if this behavior should never be stopped. (See Notes 2 and 3.)
intentSelection	String	Enter “DRAW”, “HIGHEST”, or “THRESHOLD”. (See Note 4.)
threshold	double	If intentSelection is “THRESHOLD”, the threshold value in the range [0.0, 1.0].

Table 7.3.23 AgentBehaviors Worksheet/Table.

Note 1: Enter “NONE” if the “behaviorType” field in Table 7.3.17 is either “INFRASTRUCTURE” or “COMMUNICATE” for *behaviorName*; otherwise, enter the distribution according to Note 2.

Note 2: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

Note 3: If a behavior is stopped it cannot be re-started until the start of the next replication.

Note 4: The agent can use one of three methods to choose an intention node state, i.e., choose the action it will perform:

a. DRAW – Perform a probability draw.
b. HIGHEST – Choose the state with the highest probability.
c. THRESHOLD – Declare a threshold value in the range [0.0, 1.0] and all states whose probability exceeds this value will be executed.

7.3.24 Location Worksheet/Table

Defines geographic areas within the AO. Locations are referenced by agents/groups, infrastructure, and actions. These areas are assumed to be polygons. One row of data is entered for each location as shown in Table 7.3.24.

Column/Field Name	Type	Description
name	String	The name of the location.
Level	String	Determines what level the Location is in the hierarchy of locations.
class	String	The class name of a Location defined in the rucg.mas.location package. (See Note 1.)
coordinate	String	The center coordinate of this location. Required only if *class* is "HexLocation" or "rucg.mas.location.HexLocation". The format depends upon the coordinate system. (See Note 2.) Ignored if class is neither "HexLocation" nor "rucg.mas.location.HexLocation".
numberVertices	int	The number of vertices this location owns. Enter a positive value only if there is a need to find a location given a coordinate; otherwise enter zero.
vertexCoordinate1, vertexCoordinate2, etc.	String	If *numberVertices* is positive, enter the first vertex coordinate under vertexCoordinate1, the second vertex coordinate under vertexCoordinate2, and so on. The format depends upon the coordinate system. (See Note 2.) Ignored if *numberVertices* is zero.

Table 7.3.24 Location Worksheet/Table.

Note 1: The following Location classes are currently implemented:

a. AreaLocation – A coarse representation of a geographic area in the AO.
b. HexLocation – A Location represented by a hexagon. All hexagons in a grid are assumed to be regular hexagons (i.e., all sides are equal in length and all internal angles are 120°).

Note 3: Distributions are specified by the name of the distribution followed by the parameter(s) of the distribution within parentheses. Examples are Constant(10.0), Normal(100.0, 1.0), and Triangle(50, 100.0, 75.0).

7.3.30 ScriptedEffects Worksheet/Table

Defines the effects of a ScriptedAction on an agent's or group's set of beliefs, values, interests and positions on issues. Each issue (with its supporting beliefs, values and interests) is maintained in a Bayesian network and each effect is represented by a case file. One row must be filled out for each effect as shown in Table 7.3.30.

Column/Field Name	Type	Description
index	int	The index number of an action defined in Table 7.3.29. The number links this effect with the action.
issuePrototype	String	The name of the IssuePrototype defined in Table 7.3.3. This is the issue that is affected by the action referenced by *index*.
initiator	String	The name of an AgentPrototype defined in Table 7.3.11. This is the AgentPrototype that conducted the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.11. This is the AgentPrototype that receives this effect.
effectValue	double	The effect on *issuePrototype* for *receiver*. The value must be entered in the range [minSupportStrength, maxSupportStrength] provided by issuePrototype in Table 7.3.3.

Table 7.3.30 ScriptedEffects Worksheet/Table.

7.3.31 ScriptedAttitudeEffects Worksheet/Table

Defines the effects of a ScriptedAction on an agent's or group's set of beliefs, values, interests and attitudes. Each attitude (with its supporting beliefs, values and interests) is maintained in a Bayesian network and each effect is represented by a case file. One row must be filled out for each effect as shown in Table 7.3.31.

Column/Field Name	Type	Description
index	Int	The index number of an action defined in Table 7.3.29. The number links this effect with the action.
attitudePrototype	String	The name of the AttitudePrototype defined in Table 7.3.4. This is the attitude that is affected by the action referenced by *index*.
initiator	String	The name of an AgentPrototype defined in Table 7.3.11. This is the AgentPrototype that conducted the action referenced by *index*.

receiver	String	The name of an AgentPrototype defined in Table 7.3.11. This is the AgentPrototype that receives this effect.
effectValue	double	The effect on *attitudePrototype* for *receiver*. The value must be entered in the range [minSupportStrength, maxSupportStrength] provided by *attitudePrototype* in Table 7.3.4.

Table 7.3.31 ScriptedAttitudeEffects Worksheet/Table.

7.3.32 BehaviorAction Worksheet/Table

Defines external operations initiated by an agent or group based on a planned behavior. One row must be filled out for each action as shown in Table 7.3.32.

The result of a BehaviorAction (either success or failure) may have an effect on an entity's set of beliefs, values and interests that, in turn, affects that entity's positions on issues and attitudes. The effects on beliefs, values and interests that affect positions on issues are entered in the IssueActonEffects worksheet described in 7.3.33, "IssueActionEffects Worksheet/Table". The effects on beliefs, values and interests that affect attitudes are entered in the AttitudeActionEffects worksheet described in 7.3.34, "AttitudeActionEffects Worksheet/Table".

Column/Field Name	Type	Description
index	int	A number used to identify this action. This number is used with the "index" column in the IssueActionEffects and AttitudeActionEffects worksheets to link the action with its effect(s).
behaviorName	String	The name of the behavior declared in Table 7.3.17.
intentNodeState	String	The name of the state from the node representing the Intention to perform the behavior. The entries in this field and *behaviorName* must be consistent with the entries in the "behaviorName" and "intentNodeState" fields in Table 7.3.18.
actionType	String	The name of an ActionType from Table 7.3.28. This is the ActionType that best characterizes the action associated with *intentNodeState*.

Table 7.3.32 BehaviorAction Worksheet/Table.

7.3.33 IssueActionEffects Worksheet/Table

Defines the effects of a BehaviorAction on an agent's or group's set of beliefs, values, and interests. Each effect is represented by a draw from a distribution. One row must be filled out for each effect as shown in Table 7.3.33.

Column/Field Name	Type	Description

index	int	The index number of an action defined in Table 7.3.32. The number links this effect with the action.
issuePrototype	String	The name of the IssuePrototype defined in Table 7.3.3. This is the IssuePrototype that is affected by the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.11. This is the AgentPrototype that receives this effect.
consumableType	String	Match *index* from this table with the index number in Table 7.3.32 and obtain the associated “behaviorName” from Table 7.3.32. If the “behaviorType” field in Table 7.3.17 is “INFRASTRUCTURE” for “behaviorName”, enter the name of the ConsumableType that “behaviorName” is used to restock. Ignored if “behaviorType” is not “INFRASTRUCTURE”.
providerAssociation	String	Match *index* from this table with the index number in Table 7.3.32 and obtain the associated “behaviorName” from Table 7.3.32. If the “behaviorType” field in Table 7.3.17 is “INFRASTRUCTURE” for “behaviorName”, enter the name of an AgentPrototype defined in Table 7.3.11 whose “isExternal” field is “TRUE”. Ignored if “behaviorType” is not “INFRASTRUCTURE”.
outcome	String	Enter either “SUCCESS” or “FAIL”. This is the outcome of the action referenced by *index*.
effectValue	double	The effect on *issuePrototype* for *receiver*. The value must be entered in the range [minSupportStrength, maxSupportStrength] provided by issuePrototype in Table 7.3.3.

Table 7.3.33 IssueActionEffects Worksheet/Table.

7.3.34 AttitudeActionEffects Worksheet/Table

Defines the effects of a BehaviorAction on an agent’s or group’s set of beliefs, values, and interests. Each effect is represented by a draw from a distribution. One row must be filled out for each effect as shown in Table 7.3.34.

Column/Field Name	Type	Description
index	int	The index number of an action defined in Table 7.3.32. The number links this effect with the action.
attitudePrototype	String	The name of the AttitudePrototype defined in Table 7.3.4. This is the AttitudePrototype that is

		affected by the action referenced by *index*.
receiver	String	The name of an AgentPrototype defined in Table 7.3.11. This is the AgentPrototype that receives this effect.
consumableType	String	Match *index* from this table with the index number in Table 7.3.32 and obtain the associated "behaviorName" from Table 7.3.32. If the "behaviorType" field in Table 7.3.17 is "INFRASTRUCTURE" for "behaviorName", enter the name of the ConsumableType that "behaviorName" is used to restock. Ignored if "behaviorType" is not "INFRASTRUCTURE".
providerAssociation	String	Match *index* from this table with the index number in Table 7.3.32 and obtain the associated "behaviorName" from Table 7.3.32. If the "behaviorType" field in Table 7.3.17 is "INFRASTRUCTURE" for "behaviorName", enter the name of an AgentPrototype defined in Table 7.3.11 whose "isExternal" field is "TRUE". Ignored if "behaviorType" is not "INFRASTRUCTURE".
outcome	String	Enter either "SUCCESS" or "FAIL". This is the outcome of the action referenced by *index*.
effectValue	double	The effect on *attitudePrototype* for *receiver*. The value must be entered in the range [minSupportStrength, maxSupportStrength] provided by *attitudePrototype* in Table 7.3.4.

Table 7.3.34 AttitudeActionEffects Worksheet/Table.

7.3.35 SimpleActionUmpire Worksheet/Table

Provides rules for how the SimpleActionUmpire operates. Only one row of data is required. The data is described in Table 7.3.35.

Column/Field Name	Type	Description
name	String	The name of the umpire.
recipientsNoTarget	int	The number of agents to choose at random who will receive the effects of an action if that action does not specify a target.
recepientsInfra	int	The number of agents to choose at random who will receive the effects of an action if that action specifies an infrastructure target.
doNotPassInterval	double	A period of time during which an agent will only pass an action once to other agents in its social network. (See Note 1.)
sociabilityMethod	String	Enter "K_NEAREST_NEIGHBOR" or

d. For the ActionDataLogger, the PropertyName takes the form “Action-<ActionType name>”. For example, if the ActionType name is “DamageInfrastructure”, the PropertyName is “Action-DamageInfrastructure”.
e. For the BehaviorDataLogger, the PropertyName takes the form “Behavior-<Behavior type name>. For example, if the behavior type is “INSURGENT_ACTION”, the PropertyName is “Behavior-INSURGENT_ACTION”. (See 7.3.17, “Behavior Worksheet/Table” for a list of behavior types.)
f. For the HomophilyNetworkDataLogger, the PropertyName is always “Homophily-ALL“.
g. For the LocationDataLogger, the PropertyName takes the form “Location-ALL” if EntityName is “ALL”, or “Location-<entity name>” if EntityName is the name of an entity. For example, if the EntityName is “Foo”, the PropertyName is “Location-Foo”.
h. For the CountDataLogger, the PropertyName can take the following forms:
    1) “NumberServed -ALL” if EntityName is “ALL”, or “NumberServed -<entity name>” if EntityName is the name of an Infrastructure.
    2) “NumberBalked-ALL” if EntityName is “ALL”, or “NumberBalked--<entity name>” if EntityName is the name of an Infrastructure.
    3) “NumberReneged-ALL” if EntityName is “ALL”, or “NumberReneged -<entity name>” if EntityName is the name of an Infrastructure.
    4) “NumberArrived-ALL” if EntityName is “ALL”, or “NumberArrived -<entity name>” if EntityName is the name of an Infrastructure.
i. For the SimpleStatsDataLogger, the PropertyName can take the following forms:
    1) “WaitTime-ALL” if EntityName is “ALL”, or “WaitTime-<entity name>” if EntityName is the name of an InfrastructureServer.
    2) “ServiceTime-ALL” if EntityName is “ALL”, or “ServiceTime-<entity name>” if EntityName is the name of an InfrastructureServer.
    3) “SystemTime-ALL” if EntityName is “ALL”, or “SystemTime-<entity name>” if EntityName is the name of an InfrastructureServer.
j. For the TimeVaryingStatsDataLogger, the PropertyName can take the following forms:
    1) “QueueSize-ALL” if EntityName is “ALL”, or “QueueSize-<entity name>” if EntityName is the name of an InfrastructureServer.
    2) “AvailableServer-ALL” if EntityName is “ALL”, or “AvailableServer-<entity name>” if EntityName is the name of an InfrastructureServer.

Note 6: Required by the PositionChangeDataLogger, PositionAverageDataLogger, PositionTimeAverageDataLogger, CountDataLogger and TimeVaryingStatsDataLogger.
Note 7: Required by the PositionChangeDataLogger.

7.3.65 PaveInterface Worksheet/Table

Provides information needed for SIM to connect to a Planning, Adjudication, and Visualization Environment (PAVE) database. Only one row of data is required. The data is described in Table 7.3.65.

Column/Field Name	Type	Description

name	String	A name for the PaveInterface. If blank, SIM will run standalone, i.e., SIM will run without connecting to PAVE.
class	String	The class name of the PaveInterface defined in the rucg.mas.twg package. (See Note 1.)
server	String	The name of the server where the PAVE database resides.
db	String	The name of the PAVE database file including the path (either relative or full). The database is expected to be either Microsoft Access or Microsoft SQL Server.
User	String	The authorized user's name to access the PAVE database. Applies only to Microsoft SQL Server; leave blank, otherwise.
passwd	String	The password if an authorized user's name is required; otherwise, leave blank
driver	String	The class name for the driver to be used for the connection. (See Note 2.)
firstRerunPauseTime	double	The SIM time at which this CgPaveInterface pauses for the first time if SIM needs to be restarted from time zero during the exercise. If this is the very first time SIM is being run during the exercise, the value of this field should be -1.

Table 7.3.65 PaveInterface Worksheet/Table.

Note 1: Enter either "CgPaveInterface" or "rucg.mas.twg.CgPaveInterface" (without the quotes).

Note 2: Enter one of the following:

- sun.jdbc.odbc.JdbcOdbcDriver
- com.microsoft.sqlserver.jdbc.SQLServerDriver
- net.sourceforge.jtds.jdbc.Driver

7.4 Key Leader Engagement Umpire Input File.

The key leader engagement file is an XML file that controls the translation of PAVE model instructions into key leader engagement events in SIM. The sections that follow provide a description of the format of the file. A sample file is available in **Error! Reference source not found.**.

The root element of the file is the KeyLeaderEngagementUmpire. The file is inserted in the generated XML scenario file after the RoleGroup elements.

7.4.1 KeyLeaderEngagmentUmpire Element

The KeyLeaderEngagmentUmpire has one or more KLEHandler sub-elements. Its attributes are summarized in the following table.

APPENDIX L. SIM MAINTENANCE PLAN

This appendix contains the maintenance plan for SIM. With the transition of SIM from TRAC-MTRY to TRAC-WSMR, the Maintenance Plan outlines the most likely components in SIM that will require modification for future IW TWG.

Social Impact Module (SIM)
Maintenance Plan

TRADOC Analysis Center

29 September 2012

1 Introduction
This document identifies areas of the Social Impact Module (SIM) most likely to require code modifications as a result of future Tactical Wargame (TWG) requirements and provides tips to handle these modifications. This document applies only to SIM versions 2, 2a and 3.

2 Code
Within the SIM home directory are two subdirectories called *src* and *tests*. The former contains the source code of the model and the latter contains the unit test source code. All unit tests run off of the JUnit testing framework[1]. The home directory also contains an Apache Ant[2] buildfile called *build.xml* for compiling the code (see 3, "Building SIM") and NetBeans[3] project files in the directory *nbproject*.

It is anticipated that the most likely areas of modification to SIM in the near term will be to 1) the scenario database input, 2) output generated by SIM's own data loggers, and 3) input/output to the PAVE database. The focus of this document will be on these three topics.

Unless specified, the topics that follow apply to all three versions of SIM (2, 2a and 3).

2.1 Scenario Input
SIM scenario data is primarily entered in an Excel workbook where it is first converted to XML which, in turn, is read directly by SIM.

2.1.1 Excel Workbook
The conversion from Excel to XML is handled by the class `rucg.input.jdbc.JdbcToXmlMas` using the JDOM API[4] to generate the scenario XML file. This is done specifically by the method `JdbcToXmlMas.makeDocument()` where calls are made in series to a group of protected "process" methods. Each "process" method handles one or more worksheets in the workbook. Note that the order of the calls is significant, e.g., `processAgentPrototype()` should always be called before `processAgent()`. If new data requirements evolve that require a new worksheet, create a new "process" method and call it from within `makeDocument()`.

2.1.2 XML
The objects that make up SIM are instantiated from XML Elements primarily through the classes `edu.nps.trac.maker.DefaultObjectMaker`, `edu.nps.trac.maker.ObjectMakerHelper`, `edu.nps.trac.maker.EnumMaker` and their extensions in SIM's `rucg.mas.maker.*` packages. The `DefaultObjectMaker`, `ObjectMakerHelper` and `EnumMaker` are part of the NpsTracCommon library[5].

The `DefaultObjectMaker` is the base class for generating objects. Some SIM objects are generated directly from this ObjectMaker such as the IssuePrototype. In many cases, however, the `DefaultObjectMaker` has been extended to handle the specific data of the SIM object being

[1] http://www.junit.org/
[2] http://ant.apache.org/
[3] http://netbeans.org/
[4] http://www.jdom.org/
[5] https://soteria.nps.navy.mil/WebSVN/listing.php?repname=NpsTracCommon

generated. A new ObjectMaker normally overrides the methods `DefaultObjectMaker.makeObject()` and `DefaultObjectMaker.makeObjects()` and should be developed within an existing or new sub-package of the `rucg.mas.maker` package.

The `ObjectMakerHelper` works in tandem with the `DefaultObjectMaker`. It is required if the Element used to generate an object has child Elements with additional data. An example of this is the Agent Element used to generate a SIM agent. In SIM v2, v2a and v3 this Element will not have child Elements if the agent represents one of the TWG players. However, if the agent represents a stereotype in the civilian population, the Agent Element may have one or more child elements. The kind of child elements differs depending on the version of SIM. In SIM v2 the Agent Element may have one or more of the following child Elements:

- TrustEngineSend
- TrustEngineReceive
- IssueNet
- AttitudeNet
- Behavior
- Consumable
- ConsumptionLogic

In SIM v2a the Agent Element may have one or more of the following child Elements:

- TrustEngineSend
- TrustEngineReceive
- IssueNet
- Behavior
- Consumable
- ConsumptionLogic

In SIM v3 the Agent Element may have one or more of the following child Elements:

- TrustEngineSend
- TrustEngineReceive
- Behavior
- Consumable
- ConsumptionLogic

A new `ObjectMakerHelper` should override `ObjectMakerHelper.addAdditionalData()` to process the child Elements. In addition the `ObjectMakerHelper.specialElements` Set should be filled with the names of the child Elements. A convenient place to do this is in the constructor. For example, the default constructor for the v3 rucg.mas.maker.agent.AgentMakerHelper fills `specialElements` with names of the five child elements listed previously as follows:

```
public AgentMakerHelper() {
    specialElements.add("TrustEngineSend");
    specialElements.add("TrustEngineReceive");
```

```
        specialElements.add("Behavior");
        specialElements.add("Consumable");
        specialElements.add("ConsumptionLogic");
    } // end constructor
```

A new ObjectMaker requiring a new `ObjectMakerHelper` normally overrides `DefaultObjectMaker.addElement()`.

The `EnumMaker` is an extension of the `DefaultObjectMaker` and is used to generate objects based on `simkit.util.EnumBase`, for example, `rucg.mas.consumption.ConsumableType`. There are no extensions of `EnumMaker` in SIM.

When a new Element is added to the scenario XML file, its name should be mapped to an ObjectMaker. This is done in the file *objectMakerFactory.properties* located in the *src* directory. Example entries for the ConsumableType, IssuePrototype and Agent Elements are as follows:

```
ConsumableType=edu.nps.trac.maker.EnumMaker
IssuePrototype=edu.nps.trac.maker.DefaultObjectMaker
Agent=rucg.mas.maker.agent.AgentMaker
```

Note that child Elements should not be listed in this file.

In addition, if the new Element generates a `simkit.util.EnumBase` derived object, such as `rucg.mas.consumption.ConsumableType`, the Element's name should be mapped to the class of the `EnumBase`. This is done in the file *enumMaker.properties* also located in the *src* directory. For example, the entry for the ConsumableType Element would be as follows:

```
ConsumableType=rucg.mas.consumption.ConsumableType
```

2.2 Non-PAVE Directed Output

Data not directly written to PAVE are written to CSV files handled by data loggers in the `rucg.output.mas` package. The classes and other files that must be considered when a new logger is added are discussed below.

2.2.1 Class rucg.output.mas.MasDataLogger

This is the base class for all data loggers. All new loggers should extend this class.

2.2.2 Class rucg.output.mas.MasDataLoggerType

Defines the types of data loggers as an `enum` and specifies the header for each type. If the new logger is a new type, add the type to this class. If the new logger's type matches an existing type, nothing needs to be added to this class. In either case, insure that the `type` field of the new logger is set to the appropriate `MasDataLoggerType`. This can be done conveniently in the constructor and an example is shown below for the `PositionChangeDataLogger`:

```
    public PositionChangeDataLogger() {
        super();
        type = MasDataLoggerType.POSITION_CHANGE;
    } // end constructor
```

2.2.3 Class rucg.output.mas.MasDataLoggerFileManager

Most headers specified in `MasDataLoggerType` are fixed in terms of the number of data fields that will appear in the output regardless of the study. Some data loggers, such as the `PositionChangeDataLogger`, however, have to handle the possibility that the number of issue or belief positions may vary from study to study. For these loggers the "fixed" part of the header (i.e., the part that remains unchanged from study to study) is defined in MasDataLoggerType while the "variable" part of the header is written out in the method `MasDataLoggerFileManager.open()`. For example, the "fixed" part of the `PositionChangeDataLogger` header is entered in `MasDataLoggerType` as follows:

```
    POSITION_CHANGE("replication, time, loggerName, entityElement, "
        + "entityName, location, propertyName, mode, action, caseFile, "
        + "sendingAgent, linkWeight"),
```

The "variable" part of the header is handled by the following code block in `MasDataLoggerFileManager.open()`:

```
StringBuffer header = new StringBuffer(logger.getFileHeader());
if (logger instanceof PositionChangeDataLogger) {
    PositionablePrototype topic =
            ((PositionChangeDataLogger)logger).getTopic();
    Collection<PositionPrototype> positions = topic.getPositions();
    for (PositionPrototype pos : positions) {
        header.append(", " + pos.getName());
    }
}
```

2.2.4 Class rucg.input.jdbc.JdbcToXmlMas

This class reads a scenario Excel workbook and generates a scenario XML file for input into SIM. Data loggers are entered in the Output worksheet of the spreadsheet and are processed by `JdbcToXmlMas.processOutput()`. An XML Element is created for each logger listed in the Output worksheet. Some loggers may require that a "logOldValue" attribute is in the Element while others don't require this attribute. The following code block handles this:

```
if (!isPeriodicLogger(type) && !type.equals("ActionDataLogger")
        && !type.equals("AttitudeDataLogger")
        && !type.equals("BehaviorDataLogger")
        && !type.equals("BehaviorEffectsDataLogger")
        && !type.equals("BeliefPriorDataLogger")
        && !type.equals("CommCountDataLogger")
        && !type.equals("CommunicationDataLogger")
        && !type.equals("CountDataLogger")
        && !type.equals("CountDataSummaryLogger")
        && !type.equals("DecisionMethodDataLogger")
        && !type.equals("InfrastructureVisitDataLogger")
        && !type.equals("LocationDataLogger")
        && !type.equals("SelectActionDataLogger")
        && !type.equals("SimpleStatsDataLogger")
        && !type.equals("SimpleStatsSummaryDataLogger")
        && !type.equals("StateDataLogger")
        && !type.equals("TimeVaryingStatsDataLogger")
        && !type.equals("TimeVaryingStatsSummaryDataLogger")
        && !type.equals("ActionActivationDataLogger")) {
    String logOldValue = ResultSetUtils.getString(rs, "LogOldValue");
    if (logOldValue != null && !logOldValue.trim().equals("")) {
        logOldValue = fixBoolean(logOldValue.trim());
        logger.setAttribute("logOldValue", logOldValue);
    }
```

```
}
```

If the new logger doesn't require the attribute, add an extra condition to the if statement to check for the logger; otherwise, leave the statement alone.

2.2.5 Class rucg.mas.maker.output.DataLoggerMaker

The DataLoggerMaker creates a data logger from an XML Element. Every data logger must map its corresponding XML Element name from the scenario XML file to its Class in the DataLoggerMaker's `classMap` field. This is done in the method `DataLoggerMaker.loadObjectTypes()` and an example is shown below for the `PositionChangeDataLogger`:

```
classMap.put("PositionChangeDataLogger", PositionChangeDataLogger.class);
```

2.2.6 File objectMakerFactory.properties

This file is located in the *src* directory and defines the mapping of all of the XML Element names from the scenario XML file to the appropriate ObjectMaker classes from the `rucg.mas.maker.*` packages. Note that every data logger Element needs to be mapped to the `rucg.mas.maker.output.DataLoggerMaker` and an example is shown below for the `PositionChangeDataLogger`:

```
PositionChangeDataLogger=rucg.mas.maker.output.DataLoggerMaker
```

2.3 PAVE

SIM reads from and writes to PAVE through the `rucg.mas.twg` package. The primary classes involved are listed below:

- CgAttitude – Writes to the `CG_ObservedAttitude` table.
- CgFireEvent – Reads from the `CG_Event_To_Fire` table to create SIM actions.
- CgInfrastructure – Writes to the `CG_Infrastructure_Usage` and `Zone_Content_Commodity_Change` tables.
- CgIssueStance – Writes to the `CG_IssueStance` table.
- CgPaveInterface – Obtains `TurnLength` and `CGStartDay` from the `Game_Data` table.

In particular, observed attitudes are written to PAVE's `CG_ObservedAttitude` table by the class `rucg.mas.twg.CgAttitude`. SIM v2 maintains the attitude as five probabilities and these are written to the `CGposActiveResponse`, `CGposPassiveResponse`, `CGdoNothing`, `CGnegPassiveResponse` and `CGnegActiveResponse` columns of the `CG_ObservedAttitude` table. In SIM v2a and v3, however, the attitude is a single number. At the time v2a and v3 were implemented, the `CG_ObservedAttitude` table did not have a column dedicated to either the v2a or v3 attitude. Therefore, an existing column, `CGposActiveResponse`, was used to hold this output for both versions. When a new column is created in the `CG_ObservedAttitude` table, the following SQL string in the method `CgAttitude.writeCG_ObservedAttitude()` will have to be modified:

```
String query = "INSERT INTO CG_ObservedAttitude ("
                + "Zone_ID, "
```

```
+ "Entity_ID, "
+ "EntityType, "
+ "CGposActiveResponse, "
+ "CGCurrentTurn) "
+ "Values ("
+ szAttitude.getZoneID() + ","
+ sideIDs[i] + ",'Side',"
+ szAttitude.getAggregate() + ","
+ turnCount + ")";
```

Replace `CGposActiveResponse` in the above string with the name of the new column.

3 Building SIM

As mentioned in 2, "Code Locations", the SIM home directory contains an Ant buildfile, called *build.xml*. The code can be compiled from the command line by going to the SIM home directory and typing:

```
ant compile-test
```

The following commands are available:

Command	Description
ant compile-test	Creates a *build* directory (if it doesn't exist) and compiles the code in the *src* and *tests* directories, placing the generated .class files in the *build* directory.
ant compile	Creates a *build* directory (if it doesn't exist) and compiles the code in the *src* directory only, placing the generated .class files in the *build* directory.
ant jar	Creates a *dist* directory (if it doesn't exist) and creates a jar archive of the .class files generated from the *src* directory, placing the jar file in the *dist* directory. For SIM version 2 the jar file is named sim2.jar, for version 2a it is named sim2a.jar, and for version 3 it is sim3.jar.
ant javadoc	Creates a *dist* directory (if it doesn't exist) and generates Javadocs, placing them in the *dist* directory.
ant clTest	Runs all unit tests
ant clean	Removes the *build* and *dist* directories.

REFERENCES

[1] Arnold Buss. *Discrete Event Simulation Modeling*, 2010.

[2] Walter R. Fisher. Clarifying the narrative paradigm. *Communication Monographs*, 56:56–58, 1989.

[3] TRADOC Analysis Center Methods and Research Office. Design history of the iw twg. unpublished, 2012.

[4] Headquarters Department of the Army. *Field Manual (FM) 3-24 Counterinsurgency*, December 2006.